Strategies for Multi-robot SLAM using Robot Swarms

A Project Report

submitted by

ALEEF PULIMALA

in partial fulfilment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY IN ENGINEERING DESIGN

AND

MASTER OF TECHNOLOGY IN AUTOMOTIVE ENGINEERING

DEPARTMENT OF ENGINEERING DESIGN

INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 2019

THESIS CERTIFICATE

This is to certify that the thesis titled **Strategies for Multi-robot SLAM using Robot swarms**, submitted by **Aleef Pulimala**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology** in **Engineering Design** and **Master of Technology** in **Automotive Engineering**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof.Asokan T
Professor
Dept. of Engineering Design
IIT-Madras, 600 036

Place: Chennai

Date: 5th May 2019

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to my professor Dr.Asokan.T for giving me the opportunity to work on this project and also for his invaluable guidance, support and advice during the course of this project. I am thankful for the motivation and criticism I got from the members of Robotics Lab of Engineering Design department. I thank the developers of the packages I have used and also the helpful ROS community. I also wish to express my sincere thanks to my family members and friends for their love, support and constant encouragement without which this work could not have been completed.

ABSTRACT

KEYWORDS: SLAM ; Frontier; ROS; Exploration Strategy.;Nearest Frontier Exploration, RandEx

SLAM(Simultaneous Localization And Mapping) is a vital but a challenging problem in robotics. It plays a crucial role in autonomous exploration, disaster management, search and rescue operations which involves generating a map of an unknown territory. MR-SLAM(Multi Robot SLAM) is a major advancement in mapping in which multiple robots perform SLAM through coordination and information sharing. It is a powerful tool which is faster has relatively has less chance of failure as compared to the SR-SLAM(Single Robot SLAM). It is also a more realistic application and performs better in terms of time and energy cost. Still the amount of research gone into MR-SLAM is comparatively less than that of SR-SLAM. This is the motivation behind this project.

MR-SLAM can be divided into Centralized MR-SLAM and Decentralized MR-SLAM. Centralized MR-SLAM will have a central unit which communicates to all the robots. It controls the swarm by telling each robot what to do and also receive map updates from them. A Decentralized MR-SLAM is consisted of a number of autonomous robots which coordinates and share information among themselves and completes the mapping process. Since a robot will not have the information of all other robots and their tasks, controlling and coordinating a swarm of robots in Decentralized architecture is difficult. Strategies used in these systems plays a crucial role in its efficiency.

In this project, strategies are developed in performing Centralized and Decentralized MR-SLAM. These strategies are simulated in ROS and their results are studied. Two strategies are developed in Centralized MR-SLAM which are Localization based and Image based. The first one deals with measuring transformation between each robots and fuse their local maps together using this information. The later one considers the maps as images and tries to blend them together using image processing techniques. Both approaches are compared and analyzed. The image processing techniques is developed further to implement in Decentralized system. The maps are shared and merged when to robots communicates with each other. A new algorithm for exploration in Decentralized MR-SLAM called RandEx is developed, implemented and simulated. RandEx uses a weighted random selection of all identified frontiers and assign it as next local goal for the robot while the Nearest Frontier Exploration assigns the frontier which the closest to the robot.

A comparative study is performed in the results obtained from Centralized and Decentralized strategies. The results of the two variants of Decentralized MR-SLAM has been analyzed and it has been shown that RandEx works better than the Nearest Frontier Exploration which was implemented in the first approach.

ABBREVIATIONS

ROS	Robot Operating System
SLAM	Simultaneous Localization And Mapping
RandEx	Random Exploration
MRS	Multi-Robot System
MR-SLAM	Multi-Robot Simultaneous Localization And Mapping
SR SLAM	Single-Robot Simultaneous Localization And Mapping
Rviz	ROS Visualization
NFE	Nearest Frontier Exploration

NOTATION

θ	Orientation of robot, m
x	Pose of the robot
z	observation made by robot
m	robot map
u	control input given to the robot

CHAPTER 1

INTRODUCTION

Robots are not mere fictional characters from movies anymore. They have come a long way to assisting a human and even replace humans in some tasks. Tasks which are of great difficulty or near-impossible for a human to perform, robots can do effortlessly because of their precision and accuracy in performing a task and also they are disposable when it comes to that. The influence of robotics in out life is more than we could ever imagine.

Advancement of robotics led to developments of Multi-Robot Systems(MRS). These systems works by coordinating multiple robots to perform single task or multiple tasks at the same time. There are several advantages for this kind of a system. Resistance to failure is the major one. Failure of some robots may affect the system's efficiency but the tasks can be still carried out without a total system failure. Also time taken for tasks can be significantly reduced since the work can be divided and executed in parallel. Multi-Robot system with typical robots tends to be a better option than a precisely manufactured robot with expensive systems in terms of performance and manufacturing cost. Multi-Robot systems have developed a lot to an extent of being able to simultaneously controlling 300 drones (Refer Figure 1.1) in the Olympics Winter Games PyeongChang 2018 Closing Ceremony by intel.

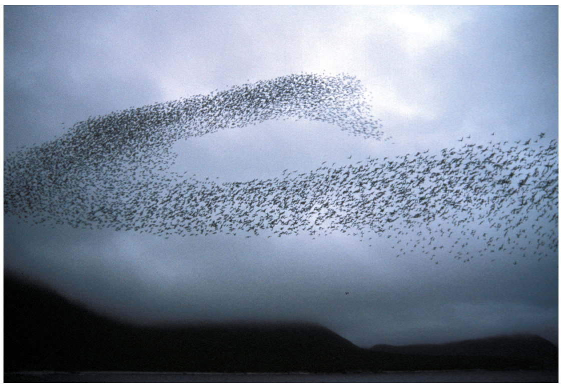
MRS is mainly classified as two. Centralized Multi-Robot System and Decentralized Multi-Robot System. Centralized System will have multiple robots being controlled and managed by a central entity. This central entity will have the control over the whole robot system. Which includes control, communication, and storage. Failure to this unit will cripple the entire system and the tasks may no longer be completed.A Centralized system will have the global information about all the robots in the system and their tasks. This makes coordinating and controlling a Centralized robot swarm easier. A decentralized system however lacks the central entity. In such systems, the whole robot system takes care of decision making, setting a local goal and local communication. The robots, because of lacking the global information takes decision based on its

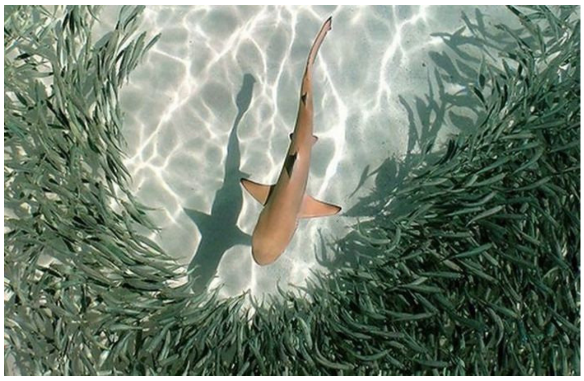
Figure 1.1: An example for Multi-Robot System. Drone show using 300 Intel Shooting Star drones for 2018 Olympics Winter Games, PyeongChang[15]. The entire fleet was controlled by one person.

limited perception of environment and it neighbours. This makes developing strategies for information sharing and exploration in Decentralized system difficult.

A robot swarm according to Brambilla et al.[3], is a multi-robot system which is autonomous, only capable of local sensing and communication, with no access to global information such as information on all robot's position and their tasks. They resemble a Decentralized Multi-Robot System. Robot swarm studies have been largely influenced by the nature and natural slams. The capabilities of these swarms as compared to the individual is fascinating. The studies based on ants, bees and termites have shown that as swarm, they were able to perform complex operations like path planning to food source, collective transport of food, collective construction, flocking of birds(Figure 1.2a) and fish schools to escape from predators(Figure 1.2b) etc. Ant Colony Optimization(ACO) solving Travelling Salesman Problem is just one of them[17].

Simultaneous Localization And Mapping (SLAM) is a powerful tool for a mobile robot to perform complex tasks in an unknown environment. It enables the robot to create a map of the surrounding while exploring. SLAM is solving two problems simultaneously. Localization of the robot in the map and creation of map according to the localization. It uses odometry data from wheel sensors and laser or radio data from environment(observations) to solve the problem. SLAM is considered as a hard problem in robotics.


(a) A flock of auklets exhibit swarm behaviour[28].


(b) Flocking behaviour of school of fish during a shark attack[22].

Multi-Robot Simultaneous Localization and Mapping (MR-SLAM) expands the SLAM into multiple robots. This is the process of performing SLAM using multiple robots which shares information to each other. Using multiple robots for SLAM have produced very promising results in total time taken for map completion and map quality [2]. Mapping process is a parallelizable process. A single robot can take a long time to complete mapping a certain area where it can be done much faster using multiple robots mapping different parts of the area simultaneously.Since it is a Multi-Robot system, a MR-SLAM system can also be divided in to Centralized MR-SLAM and Decentralized MR-SLAM. Centralized MR-SLAM will give you faster results and better control of robots while Decentralized MR-SLAM will offer you scalability across the number of robots and it has less chance of failure. Communication and exploration plays crucial role in these systems. The aim is to complete the mapping covering the least distance possible.

This project aims to develop strategies, implement and analyze them in both centralized and decentralized architectures and also proposes a new exploration algorithm called RandEx for decentralized robot swarms.

CHAPTER 2

OBJECTIVES

- Develop strategies for implementing Centralized and Decentralized Multi-Robot Simultaneous Localization And Mapping.
- Develop a new exploration strategy for Decentralized Multi-Robot system.
- Simulate the strategies in Robot Operating System.
- Analyze the results from simulations.

CHAPTER 3

LITERATURE REVIEW

This chapter gives a background study of concepts related to this project.

3.1 SLAM

SLAM was originally developed by Hugh Durrant-Whyte and John J. Leonard [9] based on earlier work by Smith, Self and Cheeseman[24]. Durrant-Whyte and Leonard originally termed it SMAL but it was later changed to give a better impact. SLAM is the ability to generate an accurate map of the environment and at the same time use the map to localize itself within it[26].Here, the robot position and the map will be unknown to the robot. SLAM uses multiple sensors to achieve this. Odometers and control inputs are used to estimate the motion of the robot and environment landmarks are sensed and mapped by the robot.

The pose of the robot in a two dimensional space is expressed by its **x** coordinate, **y** coordinate and orientation θ

$$x = (x_i, y_i, \theta_i) \tag{3.1}$$

The following represents robot pose over multiple time steps.

$$x_{1:t} = x_1, x_2, ..., x_t \tag{3.2}$$

Where x_t represents the pose at time t

A probabilistic expression is used to estimate the probability of robot pose from the previous poses and the control input.

$$p(x_t|u_t, x_{t-1})$$

This formula accounts for the uncertainties such as drift in the odometry measurement and noise sensor data for localization. This is the probability of robot pose just with the control input. Now we consider the observation by the robot. This will be the environment data robot obtain by looking around it. This data may be obtained from LIDAR, camera or radar sensors. The observation by the robot may be expressed as

$$z_{1:t} = z_1, z_2, ..., z_t \tag{3.3}$$

Here, z_t represents the observation the robot made at time t. This data might have noise or uncertainties in it. The probabilistic formulation which take into account of this uncertainty will be

$$p(z_t|m, x_t)$$

where m is the map.

Now, given the trajectory of the robot and set of observations the robot made till the time t, the SLAM problem for a single robot reduces to finding the posterior over the newly developed map(Disaanayake et al.[8]). This can be formulated as

$$p(m, x_{1:t}|z_{1:t}, u_{1:t}, x_0)$$

Here we can see that estimating map is linked with estimating trajectory. Here, the SLAM can be divided into 2. Online SLAM and Full SLAM [12]. Full SLAM needs the entire trajectory to estimate the the map while Online SLAM needs only the posterior to estimate the map. Once the accurate trajectory of the robot is known, the SLAM problem boils down to a mapping problem. i.e. The process of estimating the map based on given trajectory and observations.

$$p(m|x_{1:t}, z_{1:t})$$

If an accurate map is given, SLAM is just the process of estimating the location of the robot. i.e. a localization problem. The process of estimating the posterior on the trajectory based on map,control input and observations.

$$p(x_{1:t}|m, z_{1:t}, u_{1:t}, x_0)$$

3.2 Multi-Robot SLAM

The SLAM discussed in the previous section is a single robot SLAM. MR-SLAM is attained by improving single robot SLAM with coordination and data sharing strategy. These strategies help the robots performing single robot SLAM to coordinate with other robot, share information and merge the received information to its own map. These can be divided in the basis of their architecture. Centralized MR-SLAM and Decentralized MR-SLAM(Refer Figure3.1).

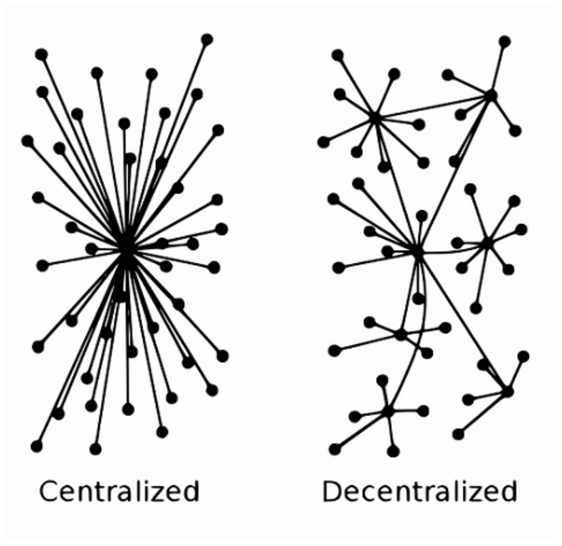


Figure 3.1: A schematic diagram of Centralized network and Decentralized network[25]

3.2.1 Centralized Multi-Robot SLAM

A centralized MR-SLAM coordinates the robot and shares data through a central entity. This central entity is responsible for controlling the robots, receiving the local map from each robot, merging them together and transmitting the global map back to the robots.

Morison et al.[21], developed a centralized MR-SLAM strategy called ***MOARSLAM*** which uses a centralized server to store the map developed by autonomous robots. The server commands the robot to explore a certain area. After exploration, the map will be transmitted to the server. If the communication fails, the robot perform autonomous mapping till the connection becomes active again. In Simmons et al.[23], the robot perform maximum likelihood estimation of map using odometry and observations. This map is transmitted to server which develop the global map from these maps. A similar strategy is implemented in Gil et al.[11], in which the robot is sending the virtual descriptors along with the odometry information to the server. The server is responsible to add these information to the global map.

The drawback of Centralized MR-SLAM is that it is heavily depended on the central entity which connects the robots and provide them global information and control. Failure of the central entity will cripple the system. Since the central server have to take care of all communication and coordination between the robots, serious scalability issues are present in this architecture. There can also be physical limitations such as actual range of communication systems used and number of robots possible for constant communication for the server due to communication channel bandwidth.

3.2.2 Decentralized Multi-Robot SLAM

Compared to centralized strategy, a decentralized system can accommodate much more number of robots to solve the given tasks. A decentralized system is highly scalable which enables the system to work with any number of robots. However not having the global information is the main disadvantage of this kind of system. Every decisions are made using local information available to each robots and this gives the robot a local goal to perform. Coordinating robots efficiently in a decentralized system can be very difficult. The two main factors of decentralized systems are communication and Task division.

Communication

In a Decentralized system, the robot develops the map by sharing information across the other robots. Howard et al.[13] introduced a method which makes the robot on meeting each other shares their sensor data. Robots use this data along with their own pose information and update the map. Another approach was by Fox et al.[10], in which robots perform SLAM as a cluster of robots which can divide in themselves if a divergent path is found. Another approach was to transmit pre-processed data of smaller size by Carlone et al.[6]. A similar approach is done by Lazaro et al.[19], by using condensed information to exchange map data between the robots. These approaches made the base for developing a variant of FastSLAM called Independent FastSLAM by Bressen et al.[5], and also accounted for noisy measurement and reduced drift by using Kalman filter.

The problem with sharing information on rendezvous is that the global information is only obtained when all the robots have communicated to each other(directly or indirectly). This put the system's ability to perform the task in some random meeting of robots. Also robot rendezvous is difficult to achieve when the environment is very large.

Task Allocation
In decentralized system, each robot gets allocated a next exploration region as a local goal because the system inherently lacks the global information. There are several exploration strategies which we will see in the next section. Since the robots have to be autonomous, every strategies have to obtain the goal using the information known to the robot such as frontiers present in the local map etc.

A decentralized swarm is large number of robot working together without the global knowledge of what the other robots are doing or have done. It is likely that the robots might repeat the work done by another robot if they have not communicated in between. If the communication about a mapped region is not happened, it will be remapped by another single robot or a cluster of robots. This reduces the efficiency of this system.

3.3 Exploration

In mobile robots, exploration is the process of navigating through an unknown region for building a representation of it. The mobile robot community have come up with many approaches to solve this problem. In multiple robot system the goal is to explore the area while reducing the amount of distance travelled by the robot. It is carried out in two phases. Identifying the most promising region for exploration and also minimizing the cost of exploration.

A frontier is the boundary between an open space and unknown space(Refer Figure3.2). By moving into frontiers, robot can explore unexplored areas. Yamaguchi et al.[29] put forward an approach for autonomous exploration using frontiers. Jose et al.[18], have developed an approach in which a scene partitioning scheme is used to assign weight to frontiers in a multi-robot system. Khalil et al.[20] came up with a hybrid decentralized coordination approach based on partially observable semi markov decision process which is a general model for enabling the robot to explore based on the local information it has and using the limited communication. Yan et al.[30] proposed another approach in which a trade based scheme is implemented in a decentralized multi-robot system so that each robot bids on a frontier based on the limited information it has. The algorithm simulates buyers and sellers in a business system to achieve dynamic task allocation.

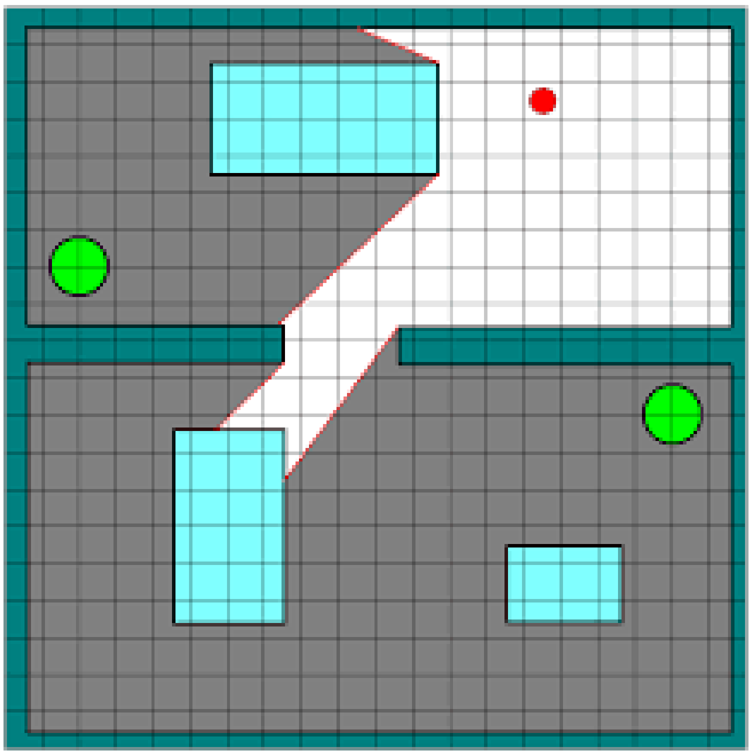


Figure 3.2: A robot(red) mapping an area[27]. Here, the red lines between dark grey(unknown) cells and white(known) cells are the frontiers.

In this project a new exploration strategy called RandEx is proposed for decentralized MR-SLAM. It introduces randomness into the frontier allocation task. More on this approach can be found in Chapter6

3.4 Map merging

Map merging is an important part in MR-SLAM. It generates global map in centralized architecture and help robots to update their local maps in decentralized architecture. Many approaches have been put forward regarding this problem.

Horner[14] proposed an image based algorithm to merge multiple maps together when overlap between them is found.It uses image processing techniques for finding similarity between maps(maps are considered as images). Once any two maps are found similar(have common region) the homography between the images are found. Using this transfomation, the images can be merged. Horner has also developed a ROS package for the same algorithm which made it very easy to implement in this project. Dinnisen et al.[7] came up with another approach in which reinforced learning is used to develop a policy to tell robot when to perform map merging. In this method, up on rendezvous, map merger is not done immediately. This reduces the chance of errors due to premature merger.

CHAPTER 4

CENTRALIZED MR-SLAM

A centralized Multi robot system needs an entity responsible for control and management of all the information the individual robots passes to it. All robots will be in constant communication with this central entity. In centralized MRSLAM, this central server will be responsible for map merging. A robot cannot perform SLAM without the central server. Failure of the central server will result in failure of the whole system.

An individual robot will send its local map to the central server whenever its map get updated. Central server will be receiving many such responses from all the robots in the systems. It collect, filters and analyze each messages and updates the global map. A robot outside the server's communication range will not be able to participate in this collective mapping. Robot went out of the server's communication range will wait for a connection form server so that they can continue cooperative SLAM.

Since all the robots are in communication with the server, the global map can be continuously updated in a predetermined frequency as if each robots are contributing to a single map. The frequency can be determined considering the transmission capability of robot-server network link, processing capability of the server and the processing power of the robots.

Every copy of the local map in each robots are interconnected through the server. So they are copies of the final map in the server. When the mapping process is completed i.e. there are no frontiers in the global map anymore,the entire robot system can be terminated via the server and the final map can be extracted from the server or from the on-board memory of any one of the robots.

4.1 Localization based Mapping Strategy

To link each robot to a single map in a global point of view and to execute cooperative SLAM, we face the following challenges.

1. Initial Robot positions. Where the robots start mapping.
2. Getting a base map. This map will initiate the whole process.
3. Linking each robot to the base map.
4. Exploration and termination

A pseudo code for Localization based Centralized MR-SLAM is given in Algorithm 1

Algorithm 1 Centralized MR-SLAM

```
one robot initializes SLAM
robot records initial map as local map
robot transmits local map to server
server receives first local map
server initializes base map with the first local map
server broadcasts the base map
while base map has frontiers do
    robot receive map from server
    if robot position is known within received map then
        robot records received map as its local map
        robot continues SLAM
        robot transmits map updates to server
    else
        robot localizes within the received map
    end
    server receives map updates
    server generates global map from current map and map updates
    server broadcasts latest map
end
server terminates mapping
```

The assumptions in this approach are,

- each robot should start within mapping range of all other robots.
- The range of central server covers the whole map

This strategy uses additive mapping from all the robots i.e. no place in the map will be mapped more than once by different robots and hence ensuring efficiency and less data duplication. Localizing robots holds the most significant cost and challenges. Successfully localizing a robot in a given map may not be always easy. It is largely depended on the surrounding of the robot. Larger the number of features in the scene, faster will be the localization. In an empty, robot will take longer to localize.

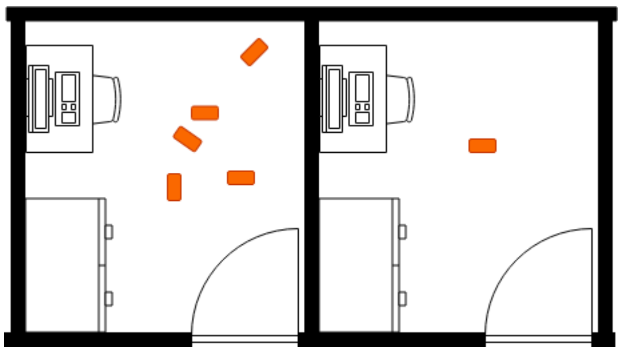


Figure 4.1: An example for wrong localization. Identical rooms allow the robot in the right room to get wrongly localized in the left room.

If a robot fails to localize, it cannot contribute to the collective mapping. It will be able to create a local map by performing SLAM. But the link between the global map and its local map is missing. This can happen because of many factors e.g. robot not finding enough features for localization or robot was not contained in the received base map(failure of first assumption) etc.

When a robot ends up with wrong localization, things can get even more complicated. A wrong localization is when a robot completes localization but identifies some other coordinates as its current position. Robot being located outside the received base map, but in an identical location to that of the base map as seen in figure 4.1, can result in wrong localization instead of not localizing.

The chances of wrong localization increases further when all robots try to localize within the limited space of received base map. As the number of localizing robots increases, the scene become more dynamic and it becomes difficult for a robot to localize. Also dynamic environment increases the probability of wrong localization.

One way to solve this problem is to localize one robot at a time. So,even if all the robots are present in the base map, as they are not moving, the environment still remain static and localization become safer and faster. But localizing this way, increases the time for the system to attain its full potential. Also by the time each robot completes localization, the base map would have developed significantly. So the time taken for the robot to travel to next un-mapped area also adds to the time cost.

After a successful localization, we get the link between the local map and the global map.i.e. The transformation between global map and local map. This transformation is applied to every map updates in local map frame and added to global map. The key factor in this process is obtaining the transformation. That is the reason why we require localization.

But if the transformation is already known, i.e. we know the link between global map and local map, the entire process becomes much simple and more resistant to failure. We acquire this by starting every robots at predetermined poses(position and orientation). We upload the robot id's and their poses in server before initializing the mapping process. Server computes the transformation between each robots and store it as initial static transformations. Using this transformation table, server can easily transform incoming map updates to global map.

This solution for localizing problem can easily be implemented in simulations as all we have to do is spawn the robots at the predetermined poses and upload the data into the server. But it is extremely tough to attain in real life tasks. The global map depends heavily on the accuracy of initial transformations. If we fail to position the robot exactly the way we intended to, The global map can become unreliable. This error will be on top the inherent errors associated with SLAM such as drift etc.

There is a way to solve the problem without the need of (explicit)transformation at all.

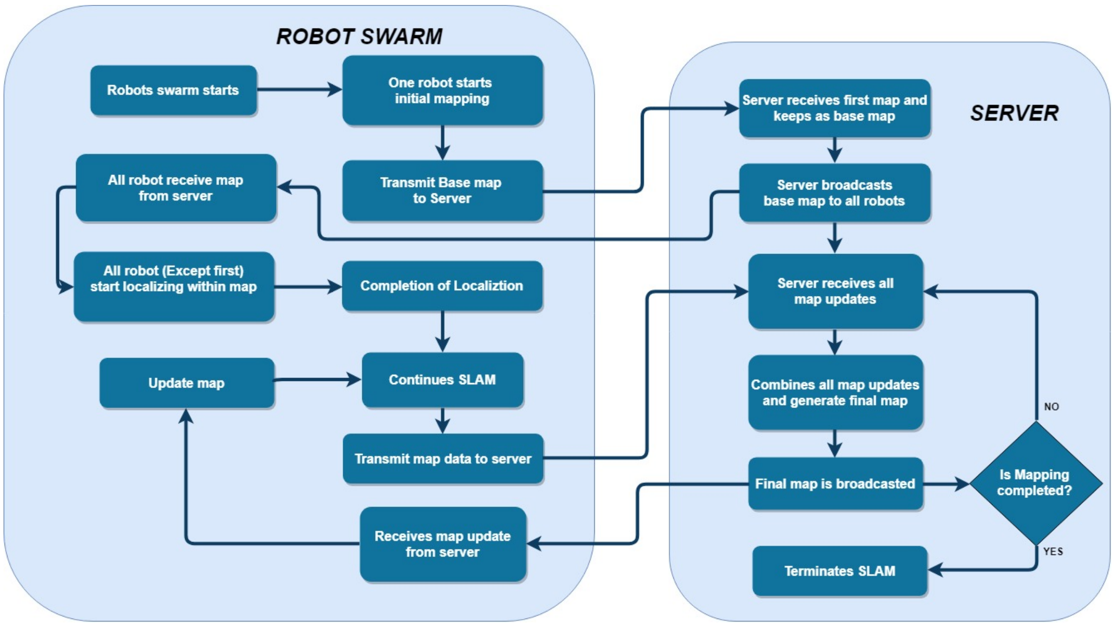

ROBOT SWARM
Robots swarm starts
One robot starts initial mapping
Transmit Base map to Server
All robot receive map from server
All robot (Except first) start localizing within map
Completion of Localiztion
Update map
Continues SLAM
Transmit map data to server
Receives map update from server
SERVER
Server receives first map and keeps as base map
Server broadcasts base map to all robots
Server receives all map updates
Combines all map updates and generate final map
Final map is broadcasted
Is Mapping completed?
NO
YES
Terminates SLAM

Figure 4.2: A flow chart showing the strategy for localization based centralized multi-robot SLAM.

4.2 Image based Mapping Strategy

In this strategy, we are eliminating the need for localization by making use of image aspects of map for stitching each map together. In this we are never making use of the transformation between robots. for this same reason, we do not need to start all robot in the mapping range of the robot which produces the initial map.

There should be enough overlap between maps for stitching them together. The steps of image stitching using a pair of images are given below

- Identify Key-points in both the maps along with its descriptors. Descriptors are useful in identifying key-points from different images. If there are many elements in the map, many key-points can be identified.
- Comparing and Matching the key points. Cycling through the key-points and matching the similar points. This may include several outliers. Which are wrongly matched points.
- Filtering outliers. Finding out the matches which varies significantly from others.Refer Figure 4.3.
- Calculating the homography transformation between the maps from the inlier matches. We need at least 8 correct matches to calculate the homography matrix. This transformation defines how one map can be pasted over the other map.
- warp one map according to the homography transformation.
- Blend two maps together.

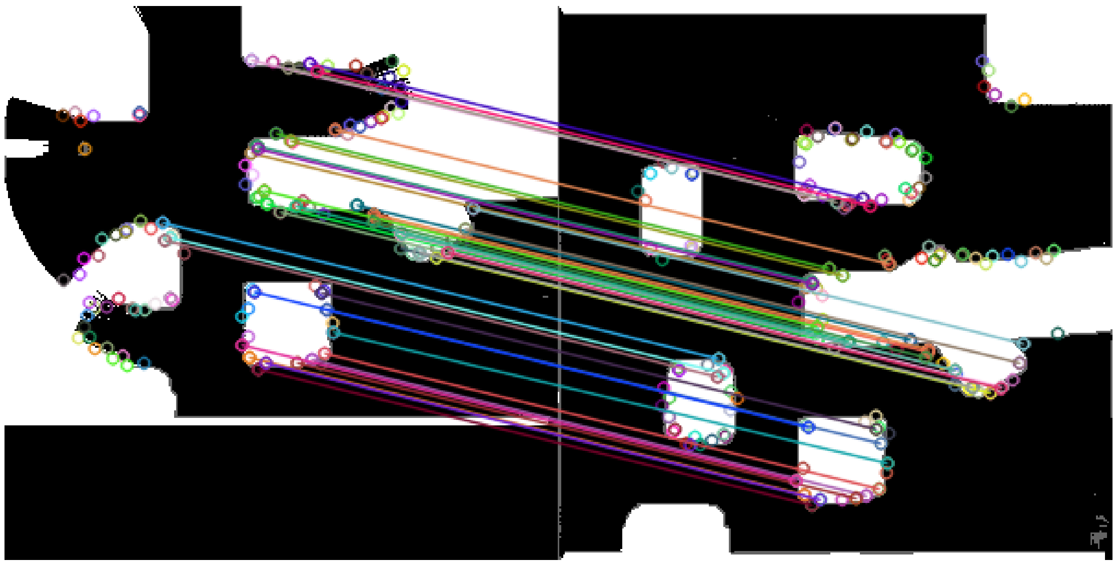


Figure 4.3: Matches between two maps after filtering outliers. Each color lines represents a match

Key-points are considered as the point of interest in an image. They can be detected even if the image has undergone affine transformations. A keypoint and descriptor defines a feature. In a map, more unique features gives better key-points and thus more reliable matches. This algorithm will work better when the environment has more edges, corners etc.

A pseudo code for Image based Centralized MR-SLAM is given in Algorithm 2

Algorithm 2 Image Based Centralized MR-SLAM

```
robot initializes SLAM
while biggest map on server has frontiers do
    robot perform SLAM and records local map
    robot transmit local map to server
    server receives local map from all robots
    if maps have sufficient overlap information between them then
        server generate a map stitching the map which has overlap
        server updates both local map to the stitched map
        server transmit the updated local map to the respective robots
    end
    robot receives updated map
end
server terminates mapping
```

Although this strategy can be faster because we are skipping the need of localization, there are some drawbacks for this approach as well.

1. We need map overlap for the algorithm to work. When the mapping is completed, there will be maps which has common regions. The size of this common region will depend on the environment type and how much information is available to make a successful stitch. The common areas implies that the same area has been remapped by multiple robots. This increases the time cost of scanning that region and also opportunity cost associated with repeat of information.

2. Two robots being in communication range does not guarantee map merger. The maps should have enough overlapping information for the merger.

3. same area can get mapped more than once. This may cause discrepancies in the final merged map.

4. This approach fails in a dynamic environment. As the process is not continuously adding features to a global map but merging parts of map together, it can fail to merge under dynamic environment. Since there can be differences in the map, the confidence factor has to be set lower which increases the chance of wrong map getting merged.

5. Server has to manage many map fragments before it can merge them together. Causes more memory usage.

As compared to Localization based approach, this approach do not need explicit time to calculate the transformations between robots through localization. It can start the mapping process right away and homography transformation becomes known as the mapping process goes on. Once a homography is found out between two map fragments from different robots, their subsequent map updates can be directly applied.

Since the robots no longer need to start within each other robot's mapping range, robots can be distributed strategically in the map so that they travel less distance and meet together in one common region which completes the map. Robot's initial position can affect the performance in this approach.

The algorithm has a closed loop at the robot's end. i.e. Robots do not require server's intervention for them to perform mapping. Server does makes it easier and faster for robots to execute the task. So the robots can continue mapping without checking if the server has transmitted data. As soon as a merger is found, server updates the corresponding robots map.

So, This method uses a parallel architecture between robot and server as compared to serial architecture in the localization based approach.

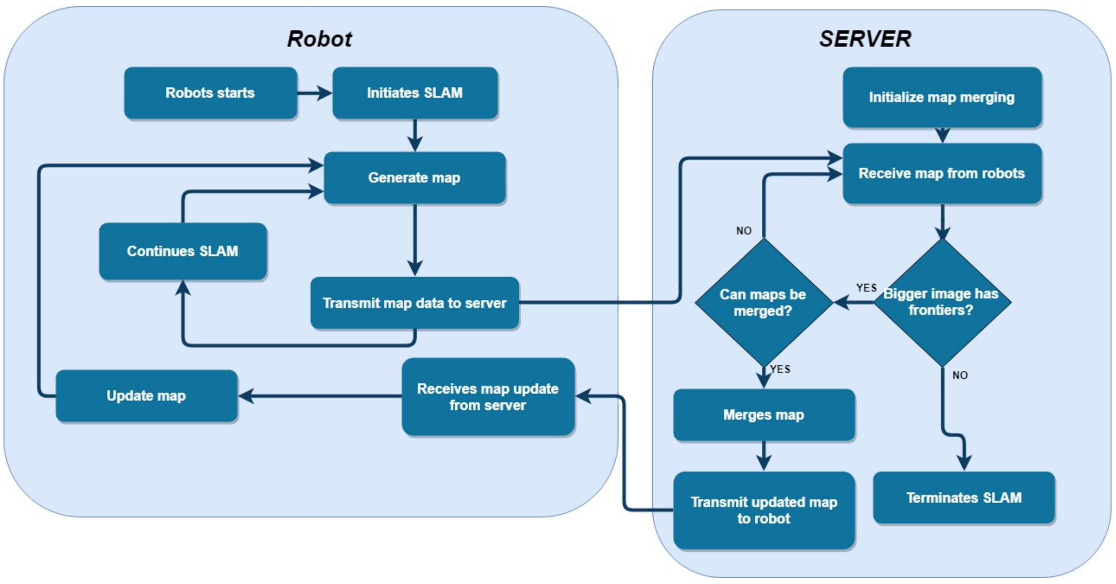


Figure 4.4: A flow chart showing the strategy for Image based centralized multi-robot SLAM.

4.3 Simulation and Analysis

Simulations are carried out using ROS. Stage is used to simulate the environment and the robots. Maps are monitored in RVIZ. Navigation2D[16] package is used for other robot functions such as navigation, obstacle avoidance, SLAM, etc. Google cartographer is the SLAM package used in the simulations. Cave map is the selected map in this simulation. This is also called as a world in stage. the map is big enough for multi-robot mapping. The world loaded in stage simulator is given in Figure 4.5

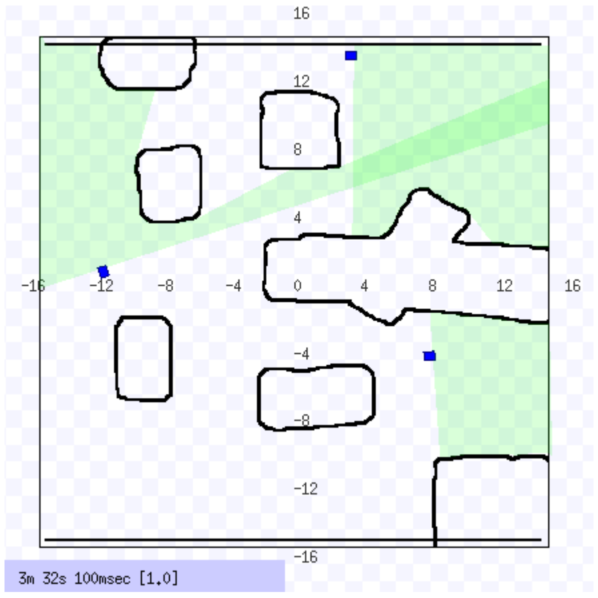


Figure 4.5: A Stage simulator loaded with cave map with 3 robots

4.3.1 Localization Based Centralized SLAM

For the localization based approach, the steps are as follows,

1. Launch ROS packages for the simulation. important packages are robot packages, stage simulator and Rviz.
2. Load the environment in stage and spawn the robots at the predefined locations.
3. Start initial mapping sequence for the first robot. This is a sequence of movements, going straight and then taking a complete rotation.
4. The initial map is synced to the other robots through the server. This will be considered as a base for global map.
5. Once other robots receive predefined area of map, A particle filter is initialized for localization.

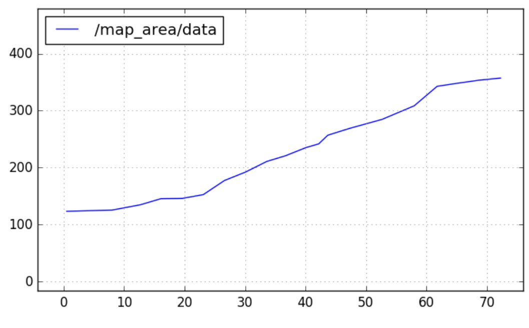
6. Localizing is performed with the particle filter. Particle filter is initialized within the received map at that specific instant. If the robot is outside the received map at that instant, the localizing will fail. This is why the assumption of every robot starts within the mapping range of every other robots very important.

7. Once every robot complete localization, they can be set into to exploration state. The Exploration strategy is Nearest frontier exploration. The robot use obstacle avoidance also during navigation to the nearest frontier.

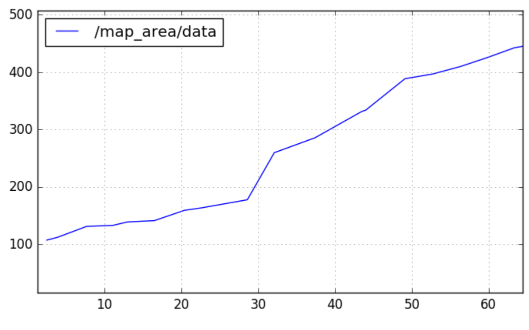
8. Each robot will update every other robot's local map with its own reading through the server. The exploration is completed when the global map no longer have frontiers to explore. Every robot terminates SLAM at this point and every robot will be having the latest global map as their local map.

Simulation is carried out iterating over the number of robots in the multi-robot system. starting with 2 and going till 4. A topic is created in ROS which will publish the global map area. This can be monitored and recorded. The actual data received is the known terrain grids in the Occupancy grid map. This data is used as an output which shows how long it took for the system to complete mapping and the trend of map area increment.

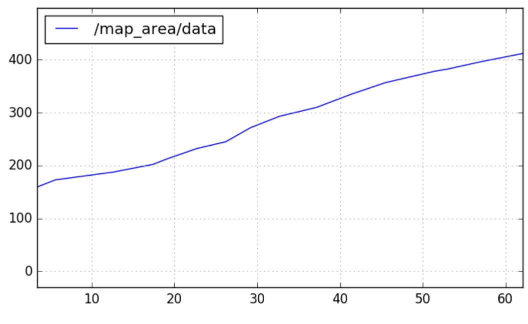
Simulation results for robots in Localization based approach.



(a) Two robots



(b) Three robots



(c) Four robots

Figure 4.6: Localization based Centralized SLAM Results

From the figure, we can see that as the number of robots increases, the map completion time decreases. This is obvious considering the fact that there are more number of robots available to perform mapping.

4.3.2 Image Based Centralized SLAM

For the Image based approach, the steps are as follows,

1. Launch ROS packages for the simulation. important packages are robot packages, stage simulator and Rviz.
2. Load the environment in stage and spawn the robots at the predefined locations.
3. Start exploration right away. The server receives the map updates from all robots. They perform SLAM independently. Nearest frontier exploration is used as exploration strategy. Obstacle avoidance is implemented while navigating to the next goal. At the same time transmits the map updates to server. Server keeps every received maps until enough overlap is found between them.
4. Server initializes image based map merger package. This will identify and merge maps together. After performing a map merger, it checks for the confidence in the merger. If the confidence falls below a preset value, the merger is dumped.
5. Whenever map merging completes, server identifies robot id's and transmit the updated maps to the corresponding robots.
6. Mapping terminates when there are no frontiers in the biggest map in server.

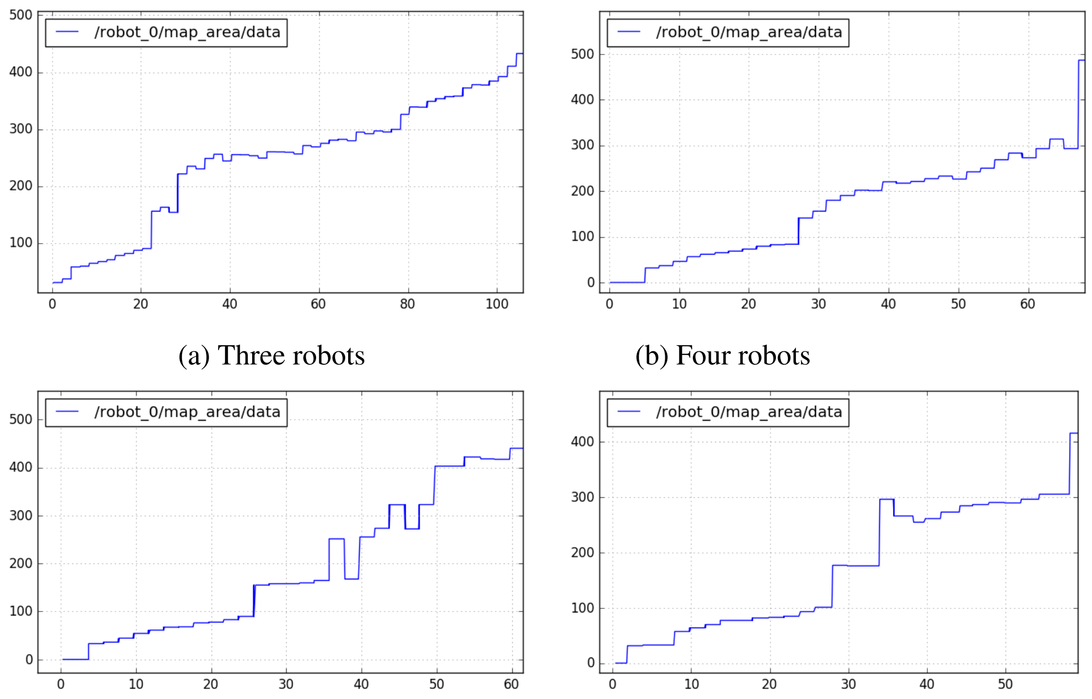
Simulation is carried out in ROS starting with 3 robots and going till 6. Although in this simulation also we are using the same map area topic, there is no single global map, the map area topic publishes the pixel count from the biggest map in the server. Since the approach is to merge two or more maps together, we can see that the graph is not smooth compared to the Localization based approach.

Simulation results for robots in Image based approach.

4.3.3 Analysis

The main difference we can see is that the shape of the map area graph. In the Localization based approach the increment in the area is constant. This happens when there is one global map and all robots contributes to that map little by little. In image based approach, we can see a step in the graph every time the server merges map.

The image based approach shows a decreasing trend just after some updates of map. This happens when the new updates went below the preset confidence level. It comes



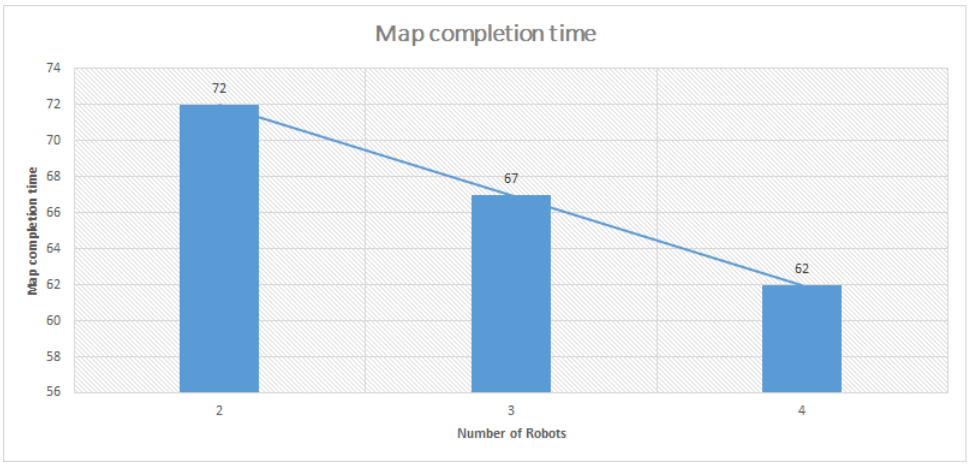
(a) Three robots (b) Four robots

(c) Five robots (d) Six robots

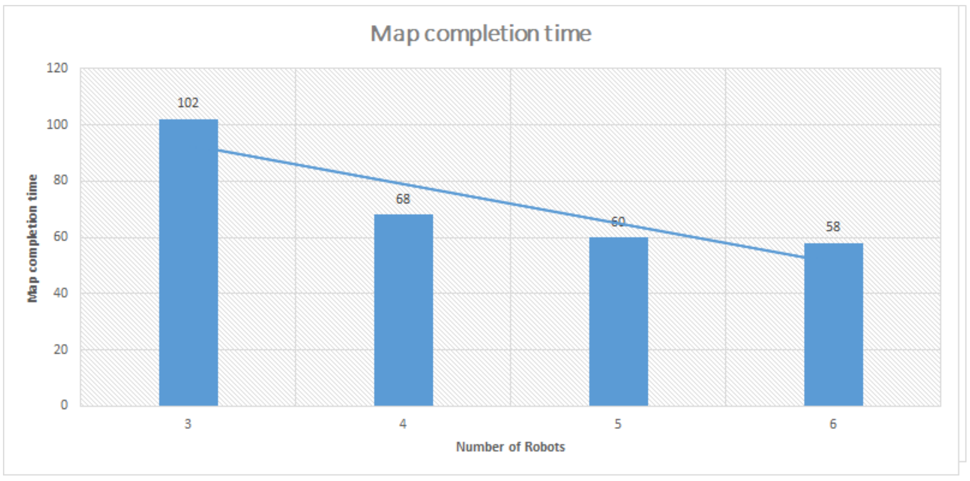
Figure 4.7: Image based Centralized SLAM Results

back to the same level when another robots produces the similar map which increases the confidence in the reading. also if another robot produces bigger fragment of the same location, we can see the reading jumps to higher value than the value it came down from.

Comparing the map completion time in Figure 4.8, we can see that the localization based approach has lower map completion time. Although it should be noted that the localization time is not considered in given graphs. The localization based graph shows the time after all the robots finishes localization.



(a) Localization based



(b) Image based

Figure 4.8: Centralized SLAM Results

CHAPTER 5

DECENTRALIZED MR-SLAM

Unlike a centralized multi-robot system, a decentralized multi robot system lacks the presence of a central entity to control robots and manage the information collected by them. In a decentralized system, each robots will be independent, autonomous and able to process the information they collect. Also they can only perform local communication with limited range.

In a decentralized system, only local communication is possible. Two robots can communicate only if they are within the communication range of each other. Robots perform independent SLAM if they are not communicating. Each robot is perfectly capable of generating the map and finish the task all by themselves. Communication lets them share the map in between so that the process become faster. While performing SLAM, every robot will be checking if it can communicate to other robots. Once the robot finds one, the SLAM process is paused, and map information is shared between them. After processing the received information, both robots can continue their individual SLAM.

For the aforementioned reasons, decentralized multi-robot SLAM is more difficult than the centralized counterpart. The absence of a global map and unavailability of robot pose information make the problem tough to solve. All exploration and map sharing rules have to be local rules coded into each robots rather than a central controlling entity.

There are some specific problems which exists only in decentralized system. like data incest, poor implementation of strategy etc. Data incest happens when a robot become over confident in its reading because it gets its own reading as different reading from other robots. Robots sending the received information back the sender causes this effect . Bressen et al.[4] developed an approach in which sub maps are used solve this issue.

Exploration strategy also plays a major role in decentralized system. The communication will only happen when two robots come in the communication range. This implies if robot never come in communication range, the whole system will be just good as single robot. An ideal exploration strategy will have both communication and also independent exploration.

But there are advantages with using decentralized system. The most important one is decentralized systems are highly scalable. Since we are only performing local coding in each robots and there are no limitations on number of robots that can be accommodated in a decentralized system, this approaches will work on any number of robots. Suppose we have 100 robots in a decentralized system, we add 100 more robots, the system efficiency will increase without any other intervention.

Since a decentralized system lacks a central entity, it has less chance of failure. In a centralized system, failure of central server causes the whole system to fail. However in decentralized system, since it can operate with any number of robots, it is more robust. There are some decentralized system with robot hierarchy. But failure of one robot will not affect other robots.

A decentralized system does not have the requirement of global coverage of communication. It only relies on local communication. So there is no concept of range of mapping for this system. Whereas in a centralized system in which the mapping range will be the communication range of the central server.

5.1 Moving on from centralized strategy

The centralized strategy from the previous chapter cannot be applied to a decentralized system mainly because of,

1. The robots are not in constant communication with other robots. Only possible mode of communication is local communication which can only take place in rendezvous.
2. The robots do not need to start within the mapping range of all other robots. They can start anywhere.
3. Localizing one robot in the other robot's map will take significant amount of time considering the fact that number of such localization will be much higher than the centralized counterpart.
4. Each robot has to be autonomous entity.

Localization based centralized approach is not suitable for a decentralized system. It will consume more time and also since the server is absent, the number of localization will also increase.

For a n robot system, with the help of a central server, the number of localizations required will be $n - 1$ .i.e. Each robot except the first perform localization. But in a decentralized system, If we are trying to localize each robot when it sees another robot for map merging, it will result in total localizations in the order of n^2 . This will increase the time required to complete the mapping process as well, because at the time of localizing, each robot has to pause the SLAM.

However, image based approach can be applied to a decentralized system considering the fact that when two robots come within communication range of each other, most probably they will be having overlapping regions in their maps. This assumption will hold as long as the mapping range is more than the communication range. When robots come within the communication range, they each will share their local maps. Once a robot receives another map, it tries to merge them together and continues the SLAM.

Another challenge associated with a decentralized system is that the robot system would not know when to terminate the SLAM process. In the cooperative mapping process, a single robot will have the completed map (a map without frontiers) at one instance of time. That robot can instantly stop SLAM right away. But other robots will still remain in the process since they have no means of knowing that the mapping process is completed by another robot until those robots come within communication range of that robot(direct or indirect). One way to solve this problem is to flood the network with a terminate signal.

We can see that a decentralized system is scalable and more resistant to failure compared to centralized system. But at the same time complexity of the system increases and developing the system has to be very careful.

5.2 Image based Decentralized Mapping Strategy

In this strategy, we only have to program robot. i.e. coming up with local rules for each robot. The algorithm is given below.

pseudo code for Image based Decentralized MR-SLAM is given in Algorithm 3. Nearest frontier exploration is used for the autonomous exploration for the robots. This

Algorithm 3 Image Based Decentralized MR-SLAM

```
robot initializes SLAM
while local map has frontiers do
    robot perform SLAM and records local map
    if Another robot is within the communication range then
        robot transmits its local map to the other robot.
        robot receives the map from other robot.
        robot initializes map merger with the two maps.
        if There is sufficient overlapping information between maps then
            robot generates merged map using the two maps.
            robot updates its local map with the new map.
        end
        robot clears the received map
    end
end
robot terminates mapping
```

exploration strategy helps in guiding robots to each other so that map merger can be performed. In Nearest frontier exploration, robots are drawn towards nearest frontiers. This increases the chance of robot-robot interactions because frontiers now act as hotspots for rendezvous. It can be when two robots explore the same frontiers or on the way to the frontiers. Also as the number of robots increases, the efficiency of the system also increases. Although the ability to scale the system up to any number of robots and being able to cover a large area without ever needing to worry about communication range makes the decentralized system a better candidate for multi robot mapping, there are some drawbacks for this method as well.

- Robot communication is not guaranteed. Even though Nearest frontier exploration partially helps, when the size of map is too big for the number of robot performing cooperative mapping, the number of encounters go down.
- Even if the robots are communicating, if the map does not have enough overlapping information, or the confidence is set too high, the map merger fails and this the system becomes less effective.
- Overlaps are essential in this approach. But at the same time they cause inefficiency due to repeated information. Amount of overlap has to be minimized.
- The robots can start SLAM from any initial positions. This influences the mapping process significantly.
- The system has a diverging characteristic, which is not a desired in an open space. So the algorithm works better at mapping an enclosed area compared to an open area.

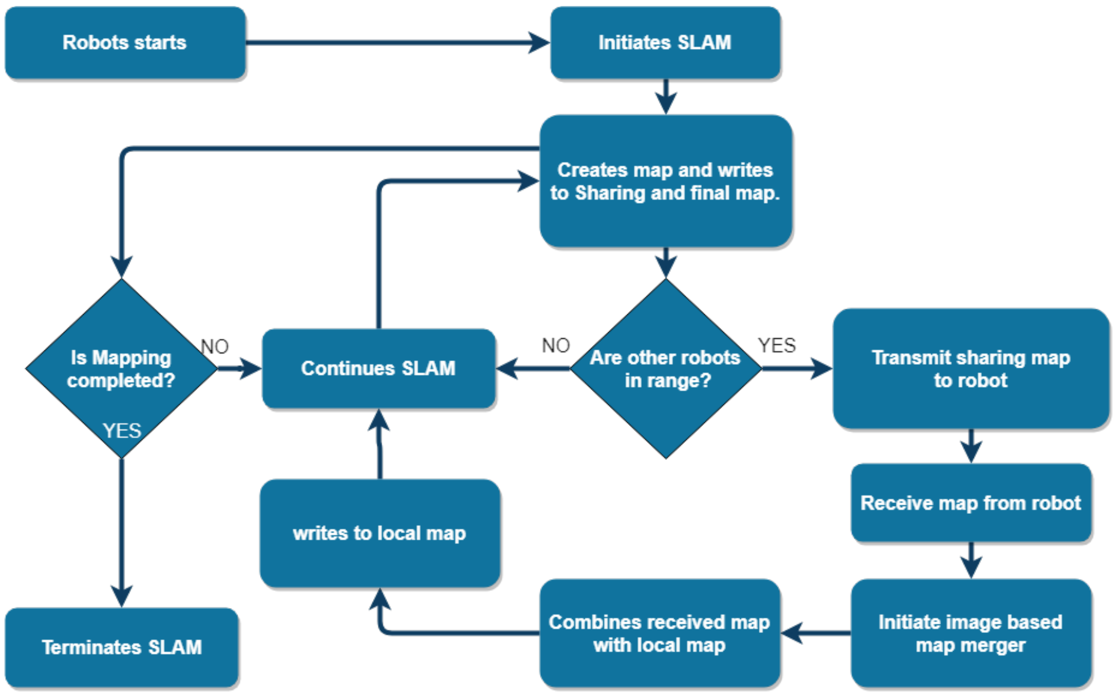

Robots starts
Initiates SLAM
Creates map and writes to Sharing and final map.
Is Mapping completed?
NO
YES
Continues SLAM
NO
Are other robots in range?
YES
Transmit sharing map to robot
Receive map from robot
writes to local map
Terminates SLAM
Combines received map with local map
Initiate image based map merger

Figure 5.1: Flow chart describing the Decentralized MR-SLAM approach

Strategically choosing the starting point will significantly reduce the time taken to complete the mapping process. Starting in a diverged state and rendezvous after covering similar distance will result in less repetition of map data and better map completion time. Robots which come close to each other can have same nearest frontiers and hence the same local goal. This causes the system to perform flocking. Flocking is the process of all robots moving together as one fleet. In mapping application, flocking is not useful as it will increase the amount of overlap significantly and will be a wastage of the capabilities of the system.

The local communication range also affects the performance. Increasing the range will increase the robot-robot interactions and thus will make the process faster. As the communication range increases, the system starts to resemble a centralized system. A centralized system can be assumed as a decentralized system with local communication range tending to infinity. As the range increases the robots perception of environment and other robots also increases.However it is not advised to increase the communication range beyond the mapping range. Communication without overlapping information in the map would not facilitate the process. Generally, increasing the communication range is a positive factor in the process. But limitation in communication range was one of the reason a decentralized system was needed in the first place. So behaviour of the system at low communication range gains more importance.

5.3 Simulation and Analysis

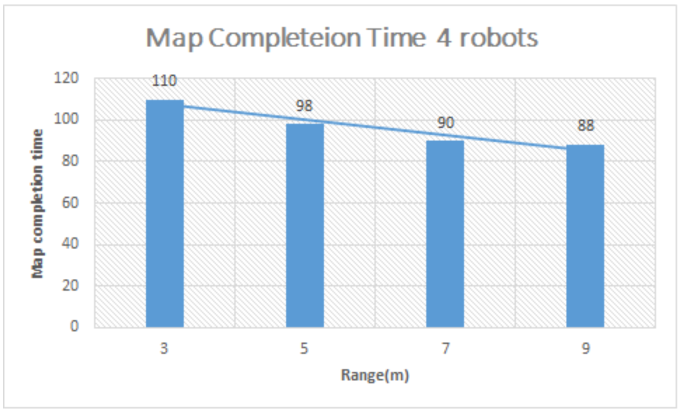
Simulations are carried out in ROS. Important packages used are Stage simulator for simulating the world, Rviz for visualizing map topics and navigation2d for other important robot packages such as obstacle avoidance, SLAM and navigation.

The simulation is iterated over two factors. **Number of Robots** and **Robot communication Range**. Number of robots varies from 3 to 6 while robot communication ranges are selected from 3m, 5m, 7m and 9m as these values collects significant variations compared to the world loaded for simulation.

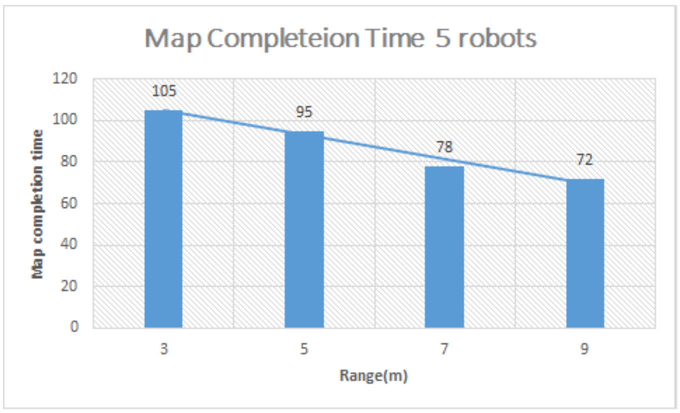
5.3.1 Analysis

The Figure 5.4 shows that the map completion time decreases as we increase the number of robots in any local communication range. The decrease in different number of robots will not be uniform as the pose of new robot also influence the total map completion time. Still we should expect lesser time when the number of robots are increasing.

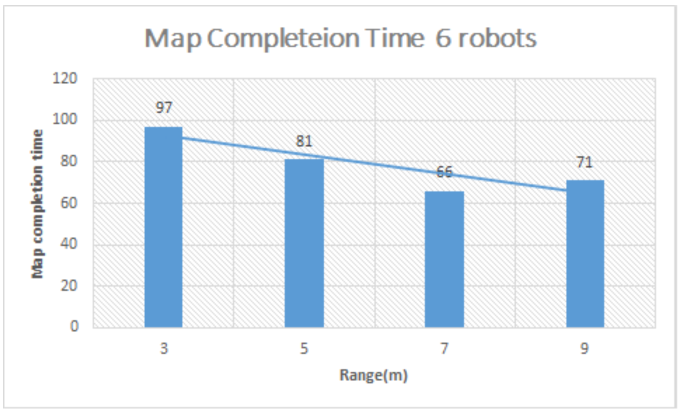
The Figure 5.5 shows as the communication range increases, the map completion time decreases. This is because increasing the range gives the robot more perception of environment and also the map merging becomes faster.



(a) Four robots

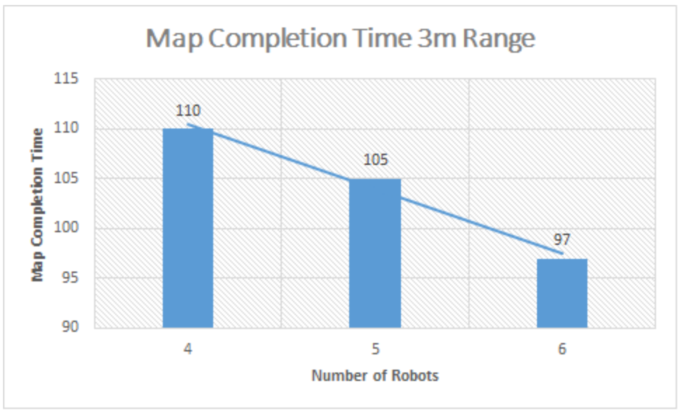


(b) Five robots

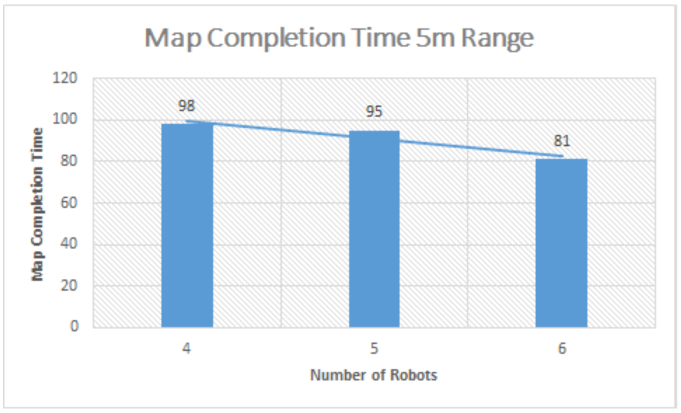


(c) Six robots

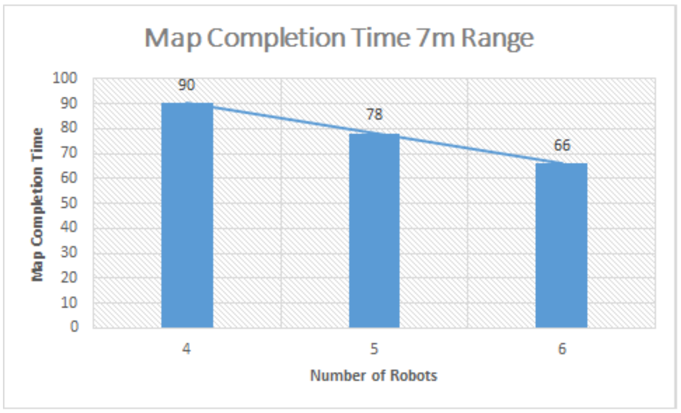
Figure 5.2: Decentralized SLAM with different number of robots



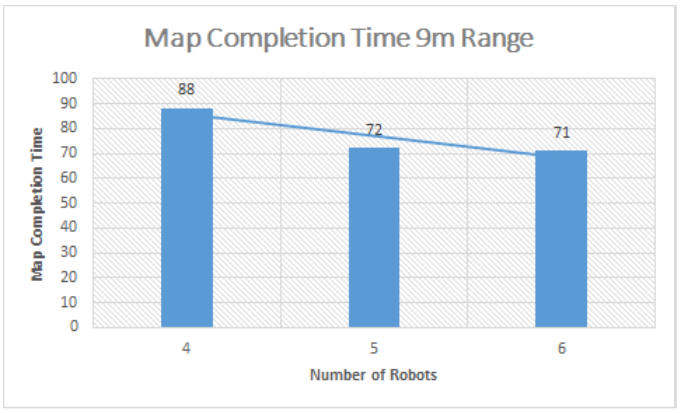
(a) 3m Communication Range



(b) 5m Communication Range



(c) 7m Communication Range



(d) 9m Communication Range

Figure 5.3: Decentralized SLAM with different communication range

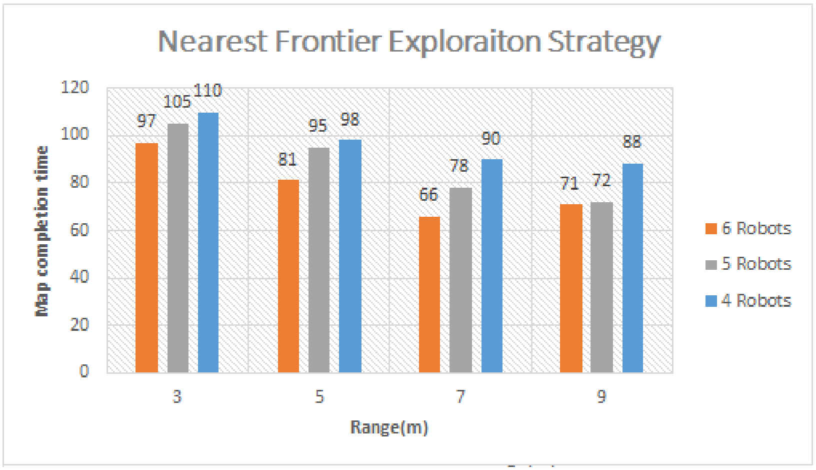


Figure 5.4: Comparison of simulation with different number of robots using nearest frontier exploration approach

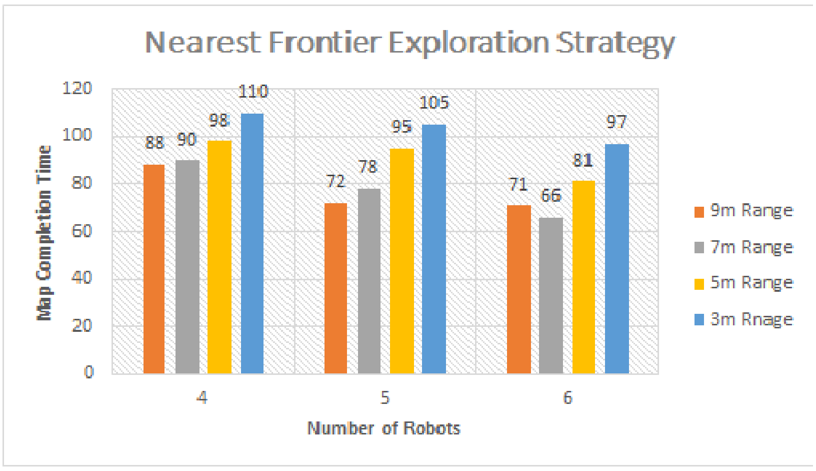


Figure 5.5: Comparison of simulation with different communication range using nearest frontier exploration approach

CHAPTER 6

RandEx ALGORITHM

RandEx(Random Exploration) is a new exploration strategy developed as a part of this project which concentrates on the exploration in decentralized multi-robot SLAM.

6.1 Background

Nearest frontier exploration works well with multi-robot system. But it was developed for autonomous exploration for a single robot. It does not take the other robots into account for cooperative mapping. This can result in poor task allocation in a decentralized system. Nearest frontier exploration searches for a frontier from the robot's position. Once it finds the first frontier, it guides the robots to the frontier location. In the process, the robot only take it's own exploration into account. Which would not matter if the robot is exploring on its own.

The problem arises when there are multiple robots doing a collaborative mapping. When to robot meets each other, the available frontiers will be the same. Nearest frontier exploration will guide both the robots to the nearest frontier. This can be same for both the robots. As stated in the previous chapter, this causes flocking in this decentralized approach, we need a diverged exploration instead of flocking. Consider 10 robots having rendezvous at one location. After that in nearest frontier exploration, All robots tends to flock together and go to the same goal.

MinPos algorithm by Bautin et al.[1] is another well known exploration strategy for centralized Multi-robot SLAM. It checks the whole map and find all frontiers. After finding each frontier points, it clusters them into different frontiers. This algorithm, take in all other robots position also. According to other robots position, the algorithm ranks each frontiers. The frontier which has highest rank is selected and the robot navigates to that location.

Here, compared to MinPos, we are lacking the pose information of other robots.

This is due to the characteristic of our system. RandEx solves the problem by bringing randomness into the process.

6.2 Methodology

RandEx algorithm has two parts. When the robots are communicating and when the robot is performing independent exploration.

While exploring independent, we can use the Nearest frontier approach as this is the most simple and effective one to implement. While performing independent SLAM, the robot only cares about expanding the map and exploring new areas. However, when another robot comes within communication range of a robot, even if map merger was successful or not,there is chance that the two robot might end up flocking together. At robot rendezvous, RandEx searches for all frontier points. It does not stop when it gets the first frontier point like Nearest frontier strategy.After getting each frontier points, they are classified into frontiers.

After storing all the frontier points, the robot has to choose one for further exploration. This process is the one which separates this algorithm from other algorithms such as Nearest frontier exploration and MinPos exploration. Since the robot do not have the information of poses of other robots, it has to make a decision based on information it has. Randomly selecting a frontier might work. Since its highly unlikely that two robot will randomly select the same frontiers and hence flocking probability decreases.

But randomly selecting a frontier can send robots to smaller frontiers which might not have good potential in exploration. So rather than taking a frontier random, RandEx takes a weighted random selection based on how big the frontiers are. Larger frontier tends to have more area to explore through them. So the weighted random selection guides the robots to largest frontiers. But since it's still a random selection, a small fraction of robots will choose the other frontiers also. Allocating more robots to large frontier will also help in better exploration.

A pseudo code for RandEx algorithm is given in Algorithm 4.

So in a way, RandEx algorithm enables Nearest frontier strategy to work with multi-

Algorithm 4 Image Based Decentralized MR-SLAM

```
Input: Map,start point of robot
Output: goal for navigation
if There are robots within communication range then
    load the current map
    load the start coordinates
    Search the map for frontier points.
    classify points based on the distance between them.
    for frontiers in the classified points do
        count the number of pixels in frontier
    end
    Perform a weighted selection of frontier using weight as frontier size
    set the selected frontier as goal
    pass the goal to navigator
else
    Perform Nearest frontier exploration
end
```

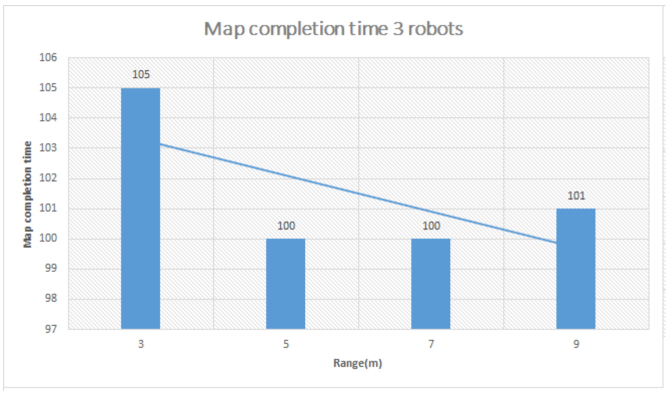
robot decentralized system.

When the number of robots are low, Nearest frontier exploration will obviously outperform RandEx algorithm because of the randomness RandEx introduced in itself. But as the number of robots increases, rendezvous tends to have more robots in it. RandEx will work better when the number of robots are higher. i.e. when there is a scope of task allocation at the time of communication.

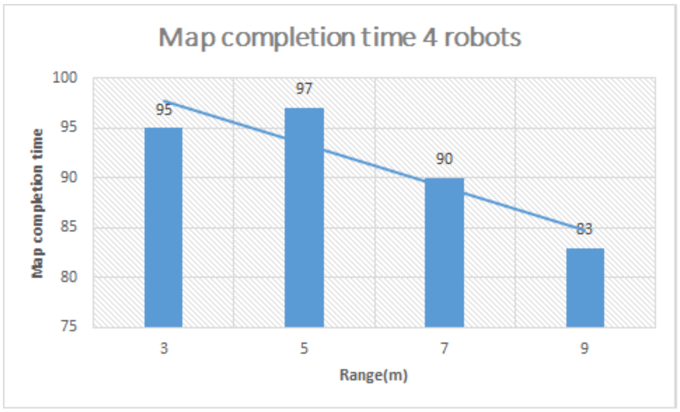
In other words, the randomness is introduced for allocating frontiers to robots. For pure exploration, it does not help if not reduce the efficiency. That is why in the algorithm itself, Nearest frontier exploration is used unless the robot is communicating with another robot.This might not be the case then the robots have large communication range. higher the range, more time the robot spends in random selection of frontiers.

6.3 Simulation and Analysis

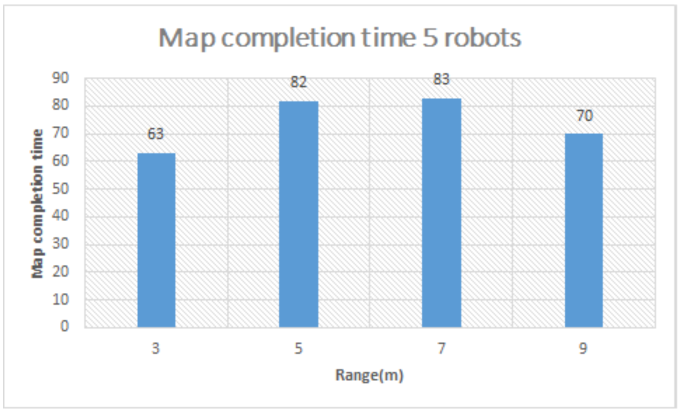
The same simulation setup for decentralized system is used for this simulation also. But instead of the Nearest frontier planner in the navigation, RandEx planner is loaded, The environment and also robot positions are also the same as decentralized simulation. Simulations are carried out for 3, 4, 5 and 6 robots and for 3m, 5m, 7m and 9m communication ranges.



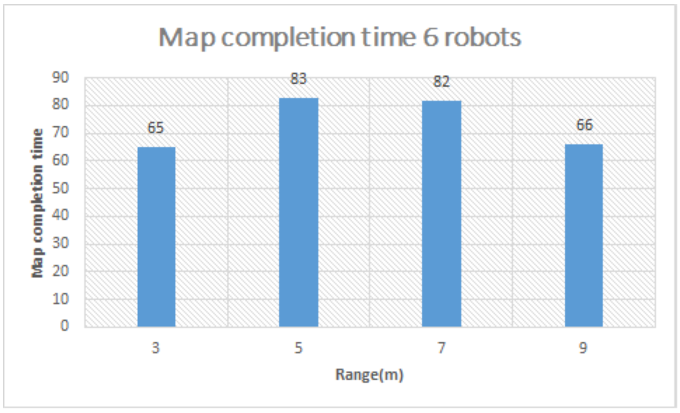
(a) Three robots



(b) Four robots

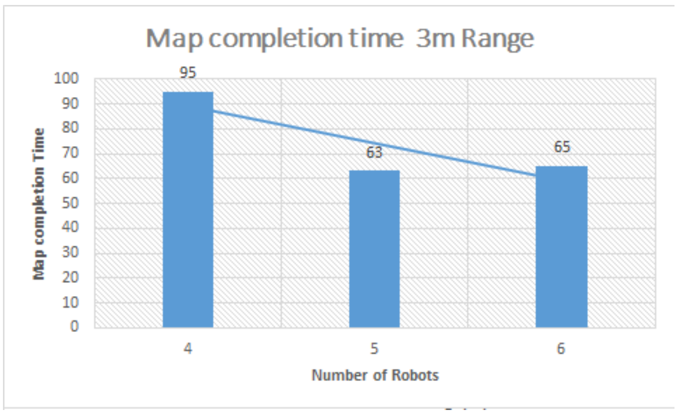


(c) Five robots

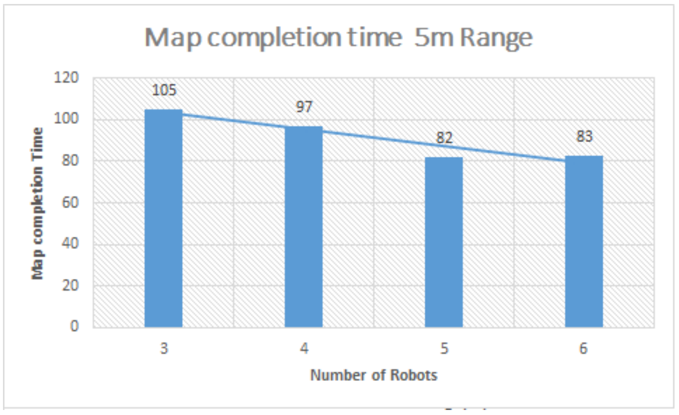


(d) Six robots

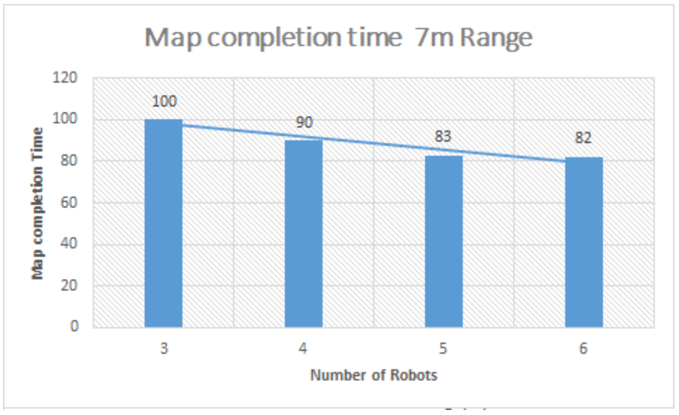
Figure 6.1: Decentralized SLAM using RandEx algorithm with different number of robots



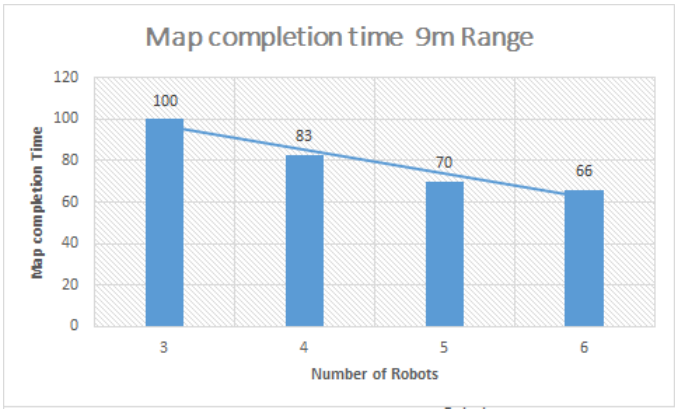
(a) 3m Communication Range



(b) 5m Communication Range



(c) 7m Communication Range



(d) 9m Communication Range

Figure 6.2: Decentralized SLAM using RandEx algorithm with different range of communication range

6.3.1 Analysis

The graphs shows a steady decrease in time as the number of robots increases. This is expected from any multi robot system. However the graph for different ranges shows an interesting results i.e. in Figure 6.2. The graph have shows low time when range is 3m but for 5m and 7m, the time increases. But when range is 9m the time again decreases. This effect become prominent in systems with more number of robots.

As discussed before, there are two factors affecting RandEx algorithm's performance, **Number of robots** and **Communication Range**. We already saw the effect on number of robots. The communication affect the performance in two ways.

Higher communication range makes the system behave like a centralized system. The perception of robot increases. Most importantly, map merging process becomes faster as the number of available maps increases.These factors favors the map building process. But in RandEx, when all robots are in communication always, the algorithm will be always trying to allocate a random frontier to the robot which reduces exploration efficiency. This time difference is noticeable when the number of robots is high. However increasing the range further can reduce the map completion time. The effect due to faster map merging overcomes the effect due to randomness. This is the reason for map completion time first increasing and then decreasing in Figure 6.2.

Figure 6.3 and 6.4 shows the effect of both communication range and number of robots. Best results are yielded at low communication range and large number of robots.

RandEx algorithm yields best results when the range is 3m. i.e. when the communication range is big enough for the robots to get allocated to frontiers but small compared to the actual map size so that the exploration is not affected by this.

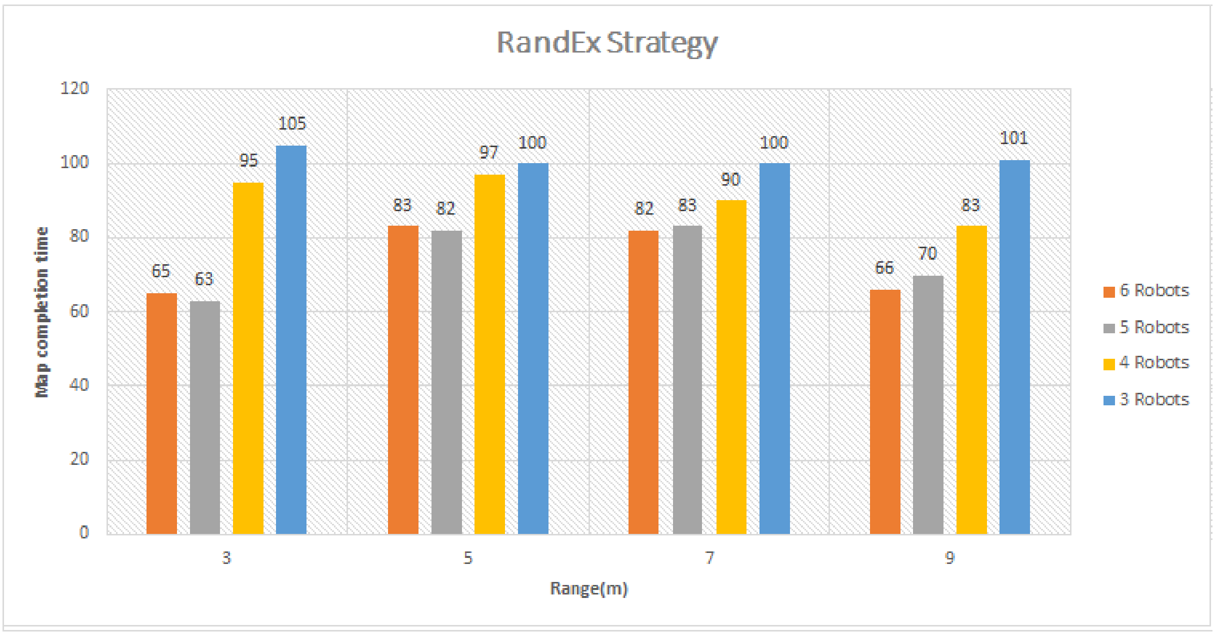


Figure 6.3: Comparison of simulation with different number of robots using RandEx approach

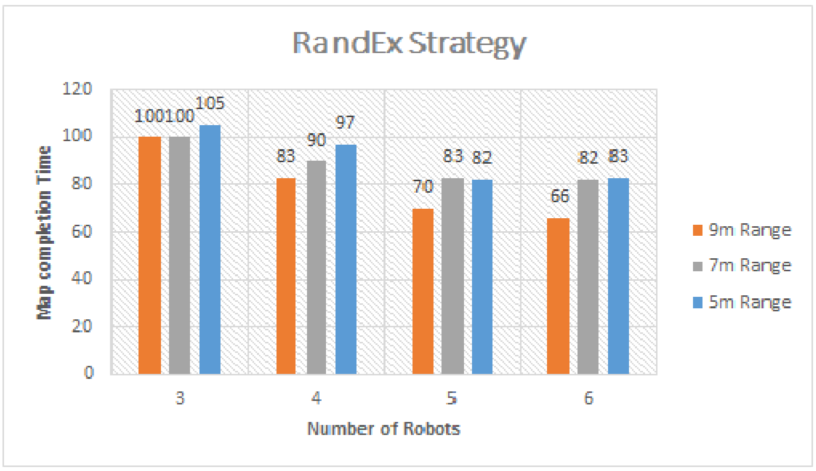


Figure 6.4: Comparison of simulation with different communication range using RandEx approach

CHAPTER 7

RESULTS AND CONCLUSIONS

Using multiple robots for mapping process increases the efficiency of the process. Time taken for both centralized system and decentralized system to completed mapping was significantly lower than Time taken for a single robot to map the same area. Refer Figure 7.1 for single robot simulation. Note that a single robot takes 145 seconds to completely map the environment. The time taken my the multi-robot systems are much lower than this value.

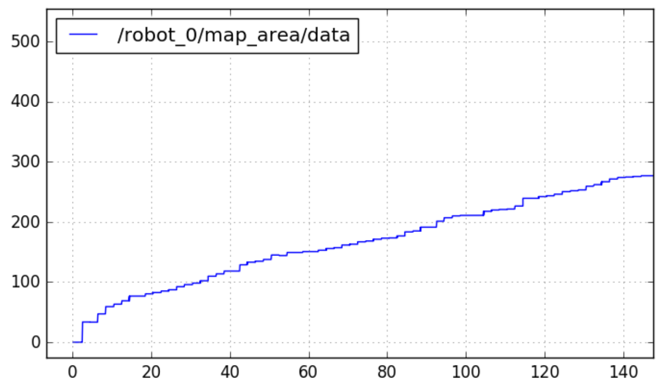


Figure 7.1: Graph showing map area with respect to time in a single robot simulation using the same environment.

In centralized approach, the time taken for complete the mapping decreased as the number of robot increased. The increase corresponding to the number of robots available in the system was a common trend in both Localization based and Image based approaches.

Using the same number of robots, Localization based approach showed better results compared to Image based. This time is measured after the localization process. But the trend of decreasing time is different in both approaches. The graph shows Image based approach works well when the number of robots are high. Even though the Localization based approach shows a steady decrease in time as the number of robots increases.

In the decentralized approach, we are doing analysis based on communication range and also the number of robots participating in the SLAM. From Figure 7.3 It is inter-

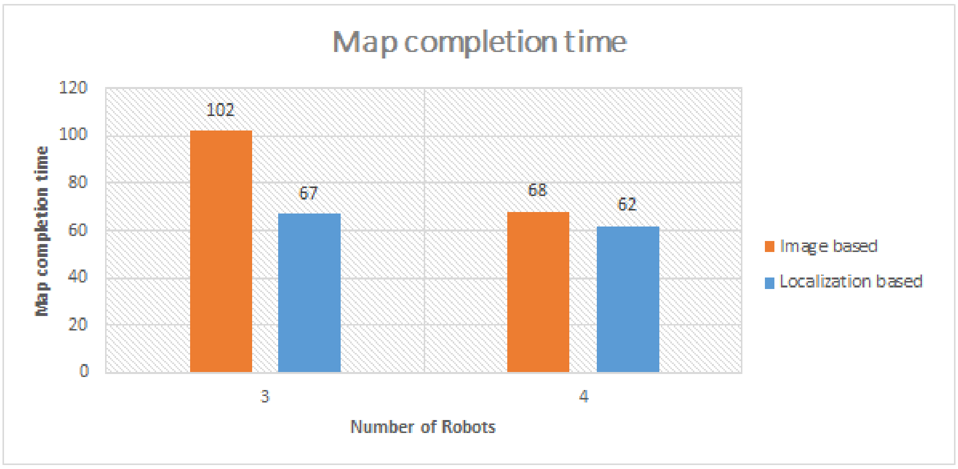


Figure 7.2: Comparison between Localization based and Image based Centralized SLAM

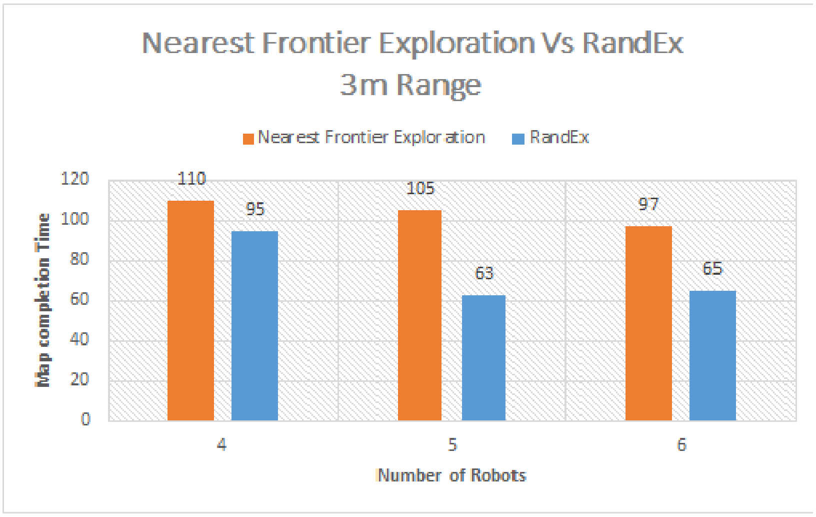


Figure 7.3: Comparison Nearest Frontier Exploration and RandEx for 3m range Note that the difference between the two algorithm increases as the number of robots increases.

esting to note that RandEx algorithm works well when the communication range is smaller. This was expected as a smaller communication range will reduce the effect of randomness in Exploration.In Almost all the test cases in this communication range, RandEx algorithm works better than Nearest Frontier Exploration algorithm.

From the Figure 7.4, we can see the trend of mapping time with respect to number of robots. When the number of robots are less, Nearest Frontier Exploration tends to work better than RandEx. This is because the effect of flocking will not be significant in a multi-robot system where the number of robots are less. So introducing randomness decreases the efficiency of the system. However the RandEx results become better when the number of robots are increased. Note that when the communication range is small,

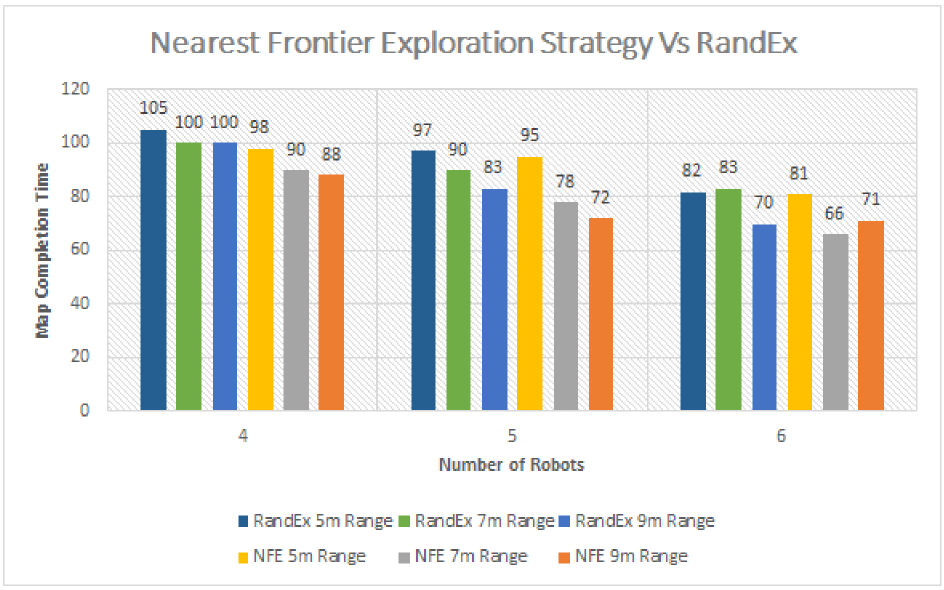


Figure 7.4: Comparison Nearest Frontier Exploration and RandEx for different communication Range and number of robots

RandEx performs better than Nearest Frontier Exploration in all the cases.

Looking at the trend of Mapping time with respect to communication range from Figure 7.5, The results become more clear. The RandEx algorithm works better than Nearest Frontier Exploraiton when the number of robots in the system is higher and the range of local communication is lower. The difference between the two algorithms in communication range of 3 meters is significant.

Since we are introducing randomness in RandEx algorithm, it may not outperform Nearest Frontier Exploration always. when the number of robots are low and the local communication range is very large, it is better to use Nearest Frontier Exploration instead of RandEx algorithm.

But the need of using a multi robot system for mapping is when the size of the environment that has to be mapped is very large compared to the robot communication range. This also implies that the local communication range as compared to the size of environment will be much lower. If it is not, we are better off using a centralized approach anyway. So, cases like these, RandEx algorithm gives the advantage of faster mapping.

Comparing the centralized and decentralized approaches, it is obvious that the centralized approach will outperform decentralized approach since it has the advantage of having global information about the map and robot positions. However, the flexibility

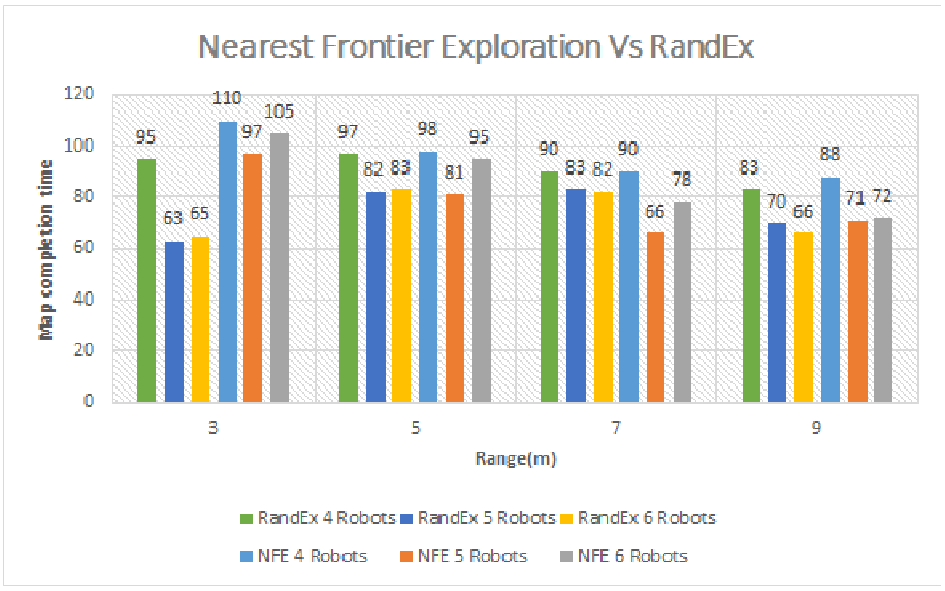


Figure 7.5: Comparison Nearest Frontier Exploration and RandEx for different communication Range and number of robots

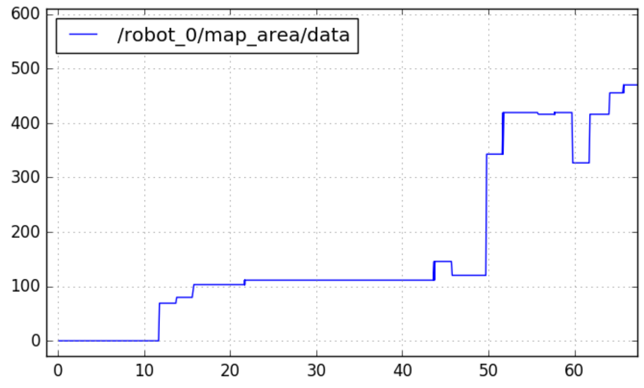


Figure 7.6: Graph between map area and time for Decentralized system with RandEx algorithm. There is 6 robot performing mapping with communication range of 3 meters. This is the lowest time taken in the whole simulation.

and scalability offered by the decentralized system make it a better strategy in many mapping tasks. Complexity is another factor of interest between these two systems. Even though coming up with local rules and programming a decentralized robot may seem complex. But producing a large number of robots like that is easier compared to centralized system. Looking at the whole system architecture, using centralized system is complex than using a decentralized system.

It is important to know about the advantages and disadvantages of all approaches in Multi Robot SLAM. Selecting one approach is depended on the environment type and size, number of robot available and also the local communication range in case of decentralized system. Every approach has its advantages and disadvantages. This project aims to help in making an informed decision.

7.1 Conclusions

After simulating the developed strategies, the following conclusions are drawn.

7.1.1 Localization based Centralized MR-SLAM

ASSUMPTIONS

- each robot should start within mapping range of all other robots.
- The range of central server covers the whole map

REMARKS

- The map completion speed of the system increases as the number of robots increases.
- Increasing robots has adverse effect on localization.
- No duplication of map data. Every part of the environment is mapped only once.
- Best choice when the number of robots and the area to be mapped are low.

7.1.2 Image based Centralized MR-SLAM

ASSUMPTIONS

- The range of central server covers the whole map.

REMARKS

- The map completion speed of the system increases as the number of robots increases.
- Initial placement of the robots plays a big role in map completion time.
- Duplication of data is present.
- Merging confidence has to be set carefully.
- Best choice when the number of robots is high and the area to be mapped is low.

7.1.3 Decentralized MR-SLAM with Nearest Frontier Exploration

ASSUMPTIONS

- Area to be mapped is a closed area.

REMARKS

- The map completion speed of the system increases as the number of robots increases.
- The map completion speed of the system increases as the local communication range of robot increases.
- Duplication of data is present.
- Merging confidence has to be set carefully.
- Best choice when the number of robots is high, the area to be mapped is high and the robot communication range is high.

7.1.4 Decentralized MR-SLAM with RandEx algorithm

ASSUMPTIONS

- Area to be mapped is a closed area.
- Local communication range is lower than mapping range.

REMARKS

- The map completion speed of the system increases as the number of robots increases.
- Duplication of data is present.
- Merging confidence has to be set carefully.
- Best choice when the number of robots is high, the area to be mapped is high and the robot communication range is low.

APPENDIX A

ROS Simulation

The entire simulation codes are uploaded into github.com for future works. See MR-SLAM or go to the url https://github.com/pplankton/MRSLAM

All the resuslts are obtained by running the above code in a system with following specifications.
CPU = Ryzen 5 2500U

RAM = 8GB DDR4

Storage = 128GB SATA III SSD

System = HP ENVY X360

Operating System = Ubuntu 16.04

Bibliography

[1] Antoine Bautin, Olivier Simonin, and François Charpillet. “MinPos : A Novel Frontier Allocation Algorithm for Multi-robot Exploration”. In: *Intelligent Robotics and Applications*. Ed. by Chun-Yi Su, Subhash Rakheja, and Honghai Liu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 496–508. ISBN: 978-3-642-33515-0.

[2] A. Birk and S. Carpin. “Merging Occupancy Grid Maps From Multiple Robots”. In: *Proceedings of the IEEE* 94.7 (July 2006), pp. 1384–1397. ISSN: 0018-9219. DOI: 10.1109/JPROC.2006.876965.

[3] Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. “Swarm Robotics: A Review from the Swarm Engineering Perspective”. In: *Swarm Intelligence* 7 (Mar. 2013), pp. 1–41. DOI: 10.1007/s11721-012-0075-2.

[4] G. Bresson, R. Aufrfffdfffdre, and R. Chapuis. “Consistent multi-robot decentralized SLAM with unknown initial positions”. In: *Proceedings of the 16th International Conference on Information Fusion*. July 2013, pp. 372–379.

[5] Guillaume Bresson, Romuald Aufrère, and Roland Chapuis. “A General Consistent Decentralized Simultaneous Localization And Mapping Solution”. In: *Robot. Auton. Syst.* 74.PA (Dec. 2015), pp. 128–147. ISSN: 0921-8890. DOI: 10.1016/j.robot.2015.07.008. URL: https://doi.org/10.1016/j.robot.2015.07.008.

[6] L. Carlone, M. Kaouk Ng, J. Du, B. Bona, and M. Indri. “Rao-Blackwellized Particle Filters multi robot SLAM with unknown initial correspondences and limited communication”. In: *2010 IEEE International Conference on Robotics and Automation*. May 2010, pp. 243–249. DOI: 10.1109/ROBOT.2010.5509307.

[7] P. Dinnissen, S. N. Givigi, and H. M. Schwartz. “Map merging of Multi-Robot SLAM using Reinforcement Learning”. In: *2012 IEEE International Confer-*

ence on Systems, Man, and Cybernetics (SMC). Oct. 2012, pp. 53–60. DOI: 10.1109/ICSMC.2012.6377676.

[8] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba. “A solution to the simultaneous localization and map building (SLAM) problem”. In: *IEEE Transactions on Robotics and Automation* 17.3 (June 2001), pp. 229–241. ISSN: 1042-296X. DOI: 10.1109/70.938381.

[9] H. Durrant-Whyte and T. Bailey. “Simultaneous localization and mapping: part I”. In: *IEEE Robotics Automation Magazine* 13.2 (June 2006), pp. 99–110. ISSN: 1070-9932. DOI: 10.1109/MRA.2006.1638022.

[10] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. “Monte Carlo Localization: Efficient Position Estimation for Mobile Robots”. In: *AAAI/IAAI*. 1999.

[11] Arturo Gil, Óscar Reinoso, Mónica Ballesta, and Miguel Juliá. “Multi-robot visual SLAM using a Rao-Blackwellized particle filter”. In: *Robotics and Autonomous Systems* 58 (2010), pp. 68–80.

[12] G. Grisetti, C. Stachniss, and W. Burgard. “Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters”. In: *IEEE Transactions on Robotics* 23.1 (Feb. 2007), pp. 34–46. ISSN: 1552-3098. DOI: 10.1109/TRO.2006.889486.

[13] Andrew Howard. “Multi-robot Simultaneous Localization and Mapping using Particle Filters”. In: *The International Journal of Robotics Research* 25.12 (2006), pp. 1243–1256. DOI: 10.1177/0278364906072250. eprint: https://doi.org/10.1177/0278364906072250. URL: https://doi.org/10.1177/0278364906072250.

[14] Jifffdfffdfffdfffd Hfffdfffdrner. *Map-merging for multi-robot system*. Bachelor’s thesis. Prague, 2016. URL: https://is.cuni.cz/webapps/zzp/detail/174125/.

[15] Intel Newsroom. *Intel Drones Go for Gold in Live Performance During 2018 Olympic Winter Games Closing Ceremony*. 2018. URL: https://simplecore.intel.com/newsroom/wp-content/uploads/sites/11/2018/02/Intel-Drone-Olympics-Closing-12.jpg.

[16] Sebastian Kasperski. *2D Navigation*. 2014. URL: `https://github.com/skasperski/navigation_2d`.

[17] B. Li, L. Wang, and W. Song. "Ant Colony Optimization for the Traveling Salesman Problem Based on Ants with Memory". In: *2008 Fourth International Conference on Natural Computation*. Vol. 7. Oct. 2008, pp. 496–501. DOI: `10.1109/ICNC.2008.354`.

[18] Jose J. Lopez-Perez, Uriel H. Hernandez-Belmonte, Juan-Pablo Ramirez-Paredes, Marco A. Contreras-Cruz, and Victor Ayala-Ramirez. "Distributed Multirobot Exploration Based on Scene Partitioning and Frontier Selection". In: *Mathematical Problems in Engineering* 2018 (2018). DOI: `10.1155/2018/2373642`. URL: `https://doi.org/10.1155/2018/2373642`.

[19] M. T. Lfffdfffdzaro, L. M. Paz, P. Pinifffdfffds, J. A. Castellanos, and G. Grisetti. "Multi-robot SLAM using condensed measurements". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Nov. 2013, pp. 1069–1076. DOI: `10.1109/IROS.2013.6696483`.

[20] Khalil Mohamed, Ayman El Shenawy, and Hany Harb. "A Hybrid Decentralized Coordinated Approach for Multi-Robot Exploration Task". In: *The Computer Journal* (Oct. 2018). ISSN: 0010-4620. DOI: `10.1093/comjnl/bxy107`. eprint: `http://oup.prod.sis.lan/comjnl/advance-article-pdf/doi/10.1093/comjnl/bxy107/26185217/bxy107.pdf`. URL: `https://doi.org/10.1093/comjnl/bxy107`.

[21] J. G. Morrison, D. Gavez-Lopez, and G. Sibley. "Scalable Multirobot Localization and Mapping with Relative Maps: Introducing MOARSLAM". In: *IEEE Control Systems Magazine* 36.2 (Apr. 2016), pp. 75–85. ISSN: 1066-033X. DOI: `10.1109/MCS.2015.2512032`.

[22] Research Gate. *Congestion Avoidance for Multiple Micro-Robots Using the Behaviour of Fish Schools*. 2012. URL: `https://www.researchgate.net/publication/289896756/figure/fig1/AS:619377544163336@1524682522910/Behaviour-of-a-school-of-fish-during-a-shark-attack.png`.

[23] Reid G. Simmons, David Apfelbaum, Wolfram Burgard, Dieter Fox, Mark Moors, Sebastian Thrun, and Håkan L. S. Younes. "Coordination for Multi-Robot Exploration and Mapping". In: *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*. AAAI Press, 2000, pp. 852–858. ISBN: 0-262-51112-6. URL: http://dl.acm.org/citation.cfm?id=647288.723404.

[24] R. Smith, M. Self, and P. Cheeseman. "Estimating uncertain spatial relationships in robotics". In: *Proceedings. 1987 IEEE International Conference on Robotics and Automation*. Vol. 4. Mar. 1987, pp. 850–850. DOI: 10.1109/ROBOT.1987.1087846.

[25] Steemit. *Decentralization is the future*. 2016. URL: https://steemitimages.com/640x0/http://www.truthcoin.info/images/cent-decent.png.

[26] Sebastian Thrun. "Robotic Mapping: A Survey". In: *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[27] Virtual Labs. *Fronteir based exploration*. 2010. URL: http://cse17-iiith.virtual-labs.ac.in/exploration/index.php?section=Theory.

[28] Wikipedia, the free encyclopedia. *Swarm Behaviour*. 2006. URL: https://upload.wikimedia.org/wikipedia/commons/5/5e/Auklet_flock_Shumagins_1986.jpg.

[29] Brian Yamauchi. "A frontier-based approach for autonomous exploration". In: *CIRA*. 1997.

[30] Zhi Yan, Nicolas Jouandeau, and Arab Ali Chérif. "Multi-robot Decentralized Exploration using a Trade-based Approach". In: *ICINCO*. 2011.